

Langmuir

Generated by Doxygen 1.8.1.2

Thu Jun 20 2013 16:24:02



# Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>LANGMUIR</b>                             | <b>1</b>  |
| <b>2</b> | <b>Namespace Index</b>                      | <b>3</b>  |
| 2.1      | Namespace List . . . . .                    | 3         |
| <b>3</b> | <b>Class Index</b>                          | <b>5</b>  |
| 3.1      | Class Hierarchy . . . . .                   | 5         |
| <b>4</b> | <b>Class Index</b>                          | <b>7</b>  |
| 4.1      | Class List . . . . .                        | 7         |
| <b>5</b> | <b>File Index</b>                           | <b>9</b>  |
| 5.1      | File List . . . . .                         | 9         |
| <b>6</b> | <b>Namespace Documentation</b>              | <b>11</b> |
| 6.1      | copy_to_frank Namespace Reference . . . . . | 11        |
| 6.1.1    | Variable Documentation . . . . .            | 11        |
| 6.1.1.1  | copy_me . . . . .                           | 11        |
| 6.1.1.2  | exclude . . . . .                           | 11        |
| 6.1.1.3  | found . . . . .                             | 11        |
| 6.1.1.4  | work . . . . .                              | 11        |
| 6.2      | Langmuir Namespace Reference . . . . .      | 11        |
| 6.2.1    | Function Documentation . . . . .            | 14        |
| 6.2.1.1  | backupFile . . . . .                        | 14        |
| 6.2.1.2  | checkSimulationParameters . . . . .         | 14        |
| 6.2.1.3  | operator<< . . . . .                        | 14        |
| 6.2.1.4  | operator<< . . . . .                        | 14        |
| 6.2.1.5  | operator<< . . . . .                        | 14        |
| 6.2.1.6  | operator<< . . . . .                        | 14        |
| 6.2.1.7  | operator<< . . . . .                        | 14        |
| 6.2.1.8  | operator<< . . . . .                        | 14        |
| 6.2.1.9  | operator<< . . . . .                        | 14        |
| 6.2.1.10 | operator<< . . . . .                        | 15        |

|          |  |           |
|----------|--|-----------|
| 6.2.1.11 | <a href="#">operator&lt;&lt;</a>                           | 15        |
| 6.2.1.12 | <a href="#">operator&lt;&lt;</a>                           | 15        |
| 6.2.1.13 | <a href="#">operator&lt;&lt;</a>                           | 15        |
| 6.2.1.14 | <a href="#">operator&lt;&lt;</a>                           | 15        |
| 6.2.1.15 | <a href="#">operator&lt;&lt;</a>                           | 15        |
| 6.2.1.16 | <a href="#">operator&gt;&gt;</a>                           | 15        |
| 6.2.1.17 | <a href="#">operator&gt;&gt;</a>                           | 15        |
| 6.2.1.18 | <a href="#">operator&gt;&gt;</a>                           | 15        |
| 6.2.1.19 | <a href="#">operator&gt;&gt;</a>                           | 15        |
| 6.2.1.20 | <a href="#">setCalculatedValues</a>                        | 15        |
| <b>7</b> | <b>Class Documentation</b>                                 | <b>17</b> |
| 7.1      | <a href="#">Langmuir::Agent Class Reference</a>            | 17        |
| 7.1.1    | <a href="#">Detailed Description</a>                       | 18        |
| 7.1.2    | <a href="#">Member Enumeration Documentation</a>           | 18        |
| 7.1.2.1  | <a href="#">Type</a>                                       | 18        |
| 7.1.3    | <a href="#">Constructor &amp; Destructor Documentation</a> | 18        |
| 7.1.3.1  | <a href="#">Agent</a>                                      | 18        |
| 7.1.3.2  | <a href="#">~Agent</a>                                     | 19        |
| 7.1.4    | <a href="#">Member Function Documentation</a>              | 19        |
| 7.1.4.1  | <a href="#">getCurrentSite</a>                             | 19        |
| 7.1.4.2  | <a href="#">getFutureSite</a>                              | 19        |
| 7.1.4.3  | <a href="#">getNeighbors</a>                               | 19        |
| 7.1.4.4  | <a href="#">getType</a>                                    | 19        |
| 7.1.4.5  | <a href="#">getWorld</a>                                   | 19        |
| 7.1.4.6  | <a href="#">setCurrentSite</a>                             | 19        |
| 7.1.4.7  | <a href="#">setFutureSite</a>                              | 19        |
| 7.1.4.8  | <a href="#">setNeighbors</a>                               | 19        |
| 7.1.4.9  | <a href="#">toQString</a>                                  | 19        |
| 7.1.5    | <a href="#">Member Data Documentation</a>                  | 20        |
| 7.1.5.1  | <a href="#">m_fSite</a>                                    | 20        |
| 7.1.5.2  | <a href="#">m_neighbors</a>                                | 20        |
| 7.1.5.3  | <a href="#">m_site</a>                                     | 20        |
| 7.1.5.4  | <a href="#">m_type</a>                                     | 20        |
| 7.1.5.5  | <a href="#">m_world</a>                                    | 20        |
| 7.2      | <a href="#">Langmuir::CarrierWriter Class Reference</a>    | 20        |
| 7.2.1    | <a href="#">Detailed Description</a>                       | 20        |
| 7.2.2    | <a href="#">Constructor &amp; Destructor Documentation</a> | 21        |
| 7.2.2.1  | <a href="#">CarrierWriter</a>                              | 21        |
| 7.2.3    | <a href="#">Member Function Documentation</a>              | 21        |

|          |  |    |
|----------|--|----|
| 7.2.3.1  | write                                  | 21 |
| 7.2.4    | Member Data Documentation              | 21 |
| 7.2.4.1  | m_stream                               | 21 |
| 7.2.4.2  | m_world                                | 21 |
| 7.3      | Langmuir::ChargeAgent Class Reference  | 21 |
| 7.3.1    | Detailed Description                   | 22 |
| 7.3.2    | Constructor & Destructor Documentation | 23 |
| 7.3.2.1  | ChargeAgent                            | 23 |
| 7.3.2.2  | ~ChargeAgent                           | 23 |
| 7.3.3    | Member Function Documentation          | 23 |
| 7.3.3.1  | bindingPotential                       | 23 |
| 7.3.3.2  | charge                                 | 23 |
| 7.3.3.3  | chooseFuture                           | 23 |
| 7.3.3.4  | compareCoulomb                         | 23 |
| 7.3.3.5  | completeTick                           | 23 |
| 7.3.3.6  | coulombCPU                             | 24 |
| 7.3.3.7  | coulombGPU                             | 24 |
| 7.3.3.8  | coulombInteraction                     | 24 |
| 7.3.3.9  | decideFuture                           | 24 |
| 7.3.3.10 | getGrid                                | 24 |
| 7.3.3.11 | getOpenCLID                            | 24 |
| 7.3.3.12 | lifetime                               | 24 |
| 7.3.3.13 | otherGrid                              | 25 |
| 7.3.3.14 | otherType                              | 25 |
| 7.3.3.15 | pathlength                             | 25 |
| 7.3.3.16 | removed                                | 25 |
| 7.3.3.17 | setOpenCLID                            | 25 |
| 7.3.3.18 | setRemoved                             | 25 |
| 7.3.4    | Member Data Documentation              | 25 |
| 7.3.4.1  | m_charge                               | 25 |
| 7.3.4.2  | m_de                                   | 25 |
| 7.3.4.3  | m_grid                                 | 26 |
| 7.3.4.4  | m_lifetime                             | 26 |
| 7.3.4.5  | m_openCLID                             | 26 |
| 7.3.4.6  | m_pathlength                           | 26 |
| 7.3.4.7  | m_removed                              | 26 |
| 7.4      | Langmuir::CheckPointer Class Reference | 26 |
| 7.4.1    | Detailed Description                   | 27 |
| 7.4.2    | Member Enumeration Documentation       | 28 |
| 7.4.2.1  | Section                                | 28 |

|          |  |    |
|----------|--|----|
| 7.4.3    | Constructor & Destructor Documentation . . . . .       | 28 |
| 7.4.3.1  | CheckPointer . . . . .                                 | 28 |
| 7.4.4    | Member Function Documentation . . . . .                | 28 |
| 7.4.4.1  | checkStream . . . . .                                  | 28 |
| 7.4.4.2  | load . . . . .   | 28 |
| 7.4.4.3  | loadDefects . . . . .                                  | 28 |
| 7.4.4.4  | loadElectrons . . . . .                                | 29 |
| 7.4.4.5  | loadFluxState . . . . .                                | 29 |
| 7.4.4.6  | loadHoles . . . . .                                    | 29 |
| 7.4.4.7  | loadParameters . . . . .                               | 29 |
| 7.4.4.8  | loadRandomState . . . . .                              | 29 |
| 7.4.4.9  | loadTrapPotentials . . . . .                           | 30 |
| 7.4.4.10 | loadTraps . . . . .                                    | 30 |
| 7.4.4.11 | save . . . . .   | 30 |
| 7.4.4.12 | saveDefects . . . . .                                  | 30 |
| 7.4.4.13 | saveElectrons . . . . .                                | 30 |
| 7.4.4.14 | saveFluxState . . . . .                                | 30 |
| 7.4.4.15 | saveHoles . . . . .                                    | 31 |
| 7.4.4.16 | saveParameters . . . . .                               | 31 |
| 7.4.4.17 | saveRandomState . . . . .                              | 31 |
| 7.4.4.18 | saveTrapPotentials . . . . .                           | 31 |
| 7.4.4.19 | saveTraps . . . . .                                    | 31 |
| 7.4.5    | Member Data Documentation . . . . .                    | 31 |
| 7.4.5.1  | m_world . . . . .                                      | 31 |
| 7.5      | Langmuir::ConfigurationInfo Struct Reference . . . . . | 32 |
| 7.5.1    | Detailed Description . . . . .                         | 32 |
| 7.5.2    | Member Data Documentation . . . . .                    | 32 |
| 7.5.2.1  | defects . . . . .                                      | 32 |
| 7.5.2.2  | electrons . . . . .                                    | 32 |
| 7.5.2.3  | fluxInfo . . . . .                                     | 32 |
| 7.5.2.4  | holes . . . . .  | 32 |
| 7.5.2.5  | trapPotentials . . . . .                               | 32 |
| 7.5.2.6  | traps . . . . .  | 33 |
| 7.6      | Langmuir::DrainAgent Class Reference . . . . .         | 33 |
| 7.6.1    | Detailed Description . . . . .                         | 33 |
| 7.6.2    | Constructor & Destructor Documentation . . . . .       | 33 |
| 7.6.2.1  | DrainAgent . . . . .                                   | 33 |
| 7.6.3    | Member Function Documentation . . . . .                | 33 |
| 7.6.3.1  | tryToAccept . . . . .                                  | 33 |
| 7.7      | Langmuir::ElectronAgent Class Reference . . . . .      | 33 |

|          |   |    |
|----------|---|----|
| 7.7.1    | Detailed Description                          | 34 |
| 7.7.2    | Constructor & Destructor Documentation        | 34 |
| 7.7.2.1  | ElectronAgent                                 | 34 |
| 7.7.3    | Member Function Documentation                 | 34 |
| 7.7.3.1  | bindingPotential                              | 34 |
| 7.7.3.2  | otherGrid                                     | 34 |
| 7.7.3.3  | otherType                                     | 35 |
| 7.8      | Langmuir::ElectronDrainAgent Class Reference  | 35 |
| 7.8.1    | Detailed Description                          | 35 |
| 7.8.2    | Constructor & Destructor Documentation        | 35 |
| 7.8.2.1  | ElectronDrainAgent                            | 35 |
| 7.8.2.2  | ElectronDrainAgent                            | 35 |
| 7.8.3    | Member Function Documentation                 | 36 |
| 7.8.3.1  | energyChange                                  | 36 |
| 7.9      | Langmuir::ElectronSourceAgent Class Reference | 36 |
| 7.9.1    | Detailed Description                          | 36 |
| 7.9.2    | Constructor & Destructor Documentation        | 36 |
| 7.9.2.1  | ElectronSourceAgent                           | 36 |
| 7.9.2.2  | ElectronSourceAgent                           | 36 |
| 7.9.3    | Member Function Documentation                 | 37 |
| 7.9.3.1  | energyChange                                  | 37 |
| 7.9.3.2  | inject  | 37 |
| 7.9.3.3  | validToInject                                 | 37 |
| 7.10     | Langmuir::ExcitonSourceAgent Class Reference  | 37 |
| 7.10.1   | Detailed Description                          | 37 |
| 7.10.2   | Constructor & Destructor Documentation        | 38 |
| 7.10.2.1 | ExcitonSourceAgent                            | 38 |
| 7.10.3   | Member Function Documentation                 | 38 |
| 7.10.3.1 | chooseSite                                    | 38 |
| 7.10.3.2 | energyChange                                  | 38 |
| 7.10.3.3 | inject  | 38 |
| 7.10.3.4 | shouldTransport                               | 38 |
| 7.10.3.5 | validToInject                                 | 38 |
| 7.11     | Langmuir::ExcitonWriter Class Reference       | 38 |
| 7.11.1   | Detailed Description                          | 39 |
| 7.11.2   | Constructor & Destructor Documentation        | 39 |
| 7.11.2.1 | ExcitonWriter                                 | 39 |
| 7.11.3   | Member Function Documentation                 | 39 |
| 7.11.3.1 | write   | 39 |
| 7.11.4   | Member Data Documentation                     | 39 |

|           |  |    |
|-----------|--|----|
| 7.11.4.1  | <code>m_stream</code>                    | 39 |
| 7.11.4.2  | <code>m_world</code>                     | 39 |
| 7.12      | Langmuir::FluxAgent Class Reference      | 39 |
| 7.12.1    | Detailed Description                     | 41 |
| 7.12.2    | Constructor & Destructor Documentation   | 41 |
| 7.12.2.1  | <code>FluxAgent</code>                   | 41 |
| 7.12.2.2  | <code>~FluxAgent</code>                  | 42 |
| 7.12.3    | Member Function Documentation            | 42 |
| 7.12.3.1  | <code>attempts</code>                    | 42 |
| 7.12.3.2  | <code>attemptsSinceLast</code>           | 42 |
| 7.12.3.3  | <code>energyChange</code>                | 42 |
| 7.12.3.4  | <code>face</code>                        | 42 |
| 7.12.3.5  | <code>faceToLetter</code>                | 42 |
| 7.12.3.6  | <code>grid</code>                        | 42 |
| 7.12.3.7  | <code>initializeSite</code>              | 42 |
| 7.12.3.8  | <code>initializeSite</code>              | 43 |
| 7.12.3.9  | <code>potential</code>                   | 43 |
| 7.12.3.10 | <code>rate</code>                        | 43 |
| 7.12.3.11 | <code>resetCounters</code>               | 43 |
| 7.12.3.12 | <code>setAttempts</code>                 | 43 |
| 7.12.3.13 | <code>setPotential</code>                | 43 |
| 7.12.3.14 | <code>setRate</code>                     | 43 |
| 7.12.3.15 | <code>setRateSmartly</code>              | 44 |
| 7.12.3.16 | <code>setSuccesses</code>                | 44 |
| 7.12.3.17 | <code>shouldTransport</code>             | 44 |
| 7.12.3.18 | <code>stepsSinceLast</code>              | 44 |
| 7.12.3.19 | <code>storeLast</code>                   | 44 |
| 7.12.3.20 | <code>successes</code>                   | 44 |
| 7.12.3.21 | <code>successesSinceLast</code>          | 45 |
| 7.12.3.22 | <code>successProbability</code>          | 45 |
| 7.12.3.23 | <code>successProbabilitySinceLast</code> | 45 |
| 7.12.3.24 | <code>successRate</code>                 | 45 |
| 7.12.3.25 | <code>successRateSinceLast</code>        | 45 |
| 7.12.4    | Member Data Documentation                | 45 |
| 7.12.4.1  | <code>m_attempts</code>                  | 45 |
| 7.12.4.2  | <code>m_face</code>                      | 45 |
| 7.12.4.3  | <code>m_grid</code>                      | 45 |
| 7.12.4.4  | <code>m_lastAttempts</code>              | 45 |
| 7.12.4.5  | <code>m_lastStep</code>                  | 45 |
| 7.12.4.6  | <code>m_lastSuccesses</code>             | 46 |



|           |  |    |
|-----------|--|----|
| 7.12.4.7  | <a href="#">m_potential</a>                                | 46 |
| 7.12.4.8  | <a href="#">m_probability</a>                              | 46 |
| 7.12.4.9  | <a href="#">m_successes</a>                                | 46 |
| 7.13      | <a href="#">Langmuir::FluxWriter Class Reference</a>       | 46 |
| 7.13.1    | <a href="#">Detailed Description</a>                       | 46 |
| 7.13.2    | <a href="#">Constructor &amp; Destructor Documentation</a> | 46 |
| 7.13.2.1  | <a href="#">FluxWriter</a>                                 | 46 |
| 7.13.3    | <a href="#">Member Function Documentation</a>              | 47 |
| 7.13.3.1  | <a href="#">write</a>                                      | 47 |
| 7.13.4    | <a href="#">Member Data Documentation</a>                  | 47 |
| 7.13.4.1  | <a href="#">m_stream</a>                                   | 47 |
| 7.13.4.2  | <a href="#">m_world</a>                                    | 47 |
| 7.14      | <a href="#">Langmuir::Grid Class Reference</a>             | 47 |
| 7.14.1    | <a href="#">Detailed Description</a>                       | 50 |
| 7.14.2    | <a href="#">Member Enumeration Documentation</a>           | 50 |
| 7.14.2.1  | <a href="#">CubeFace</a>                                   | 50 |
| 7.14.3    | <a href="#">Constructor &amp; Destructor Documentation</a> | 50 |
| 7.14.3.1  | <a href="#">Grid</a>                                       | 50 |
| 7.14.3.2  | <a href="#">~Grid</a>                                      | 51 |
| 7.14.4    | <a href="#">Member Function Documentation</a>              | 51 |
| 7.14.4.1  | <a href="#">addToPotential</a>                             | 51 |
| 7.14.4.2  | <a href="#">agentAddress</a>                               | 51 |
| 7.14.4.3  | <a href="#">agentType</a>                                  | 51 |
| 7.14.4.4  | <a href="#">getIndexS</a>                                  | 51 |
| 7.14.4.5  | <a href="#">getIndexX</a>                                  | 52 |
| 7.14.4.6  | <a href="#">getIndexY</a>                                  | 52 |
| 7.14.4.7  | <a href="#">getIndexZ</a>                                  | 52 |
| 7.14.4.8  | <a href="#">getPositionX</a>                               | 52 |
| 7.14.4.9  | <a href="#">getPositionY</a>                               | 53 |
| 7.14.4.10 | <a href="#">getPositionZ</a>                               | 53 |
| 7.14.4.11 | <a href="#">getSpecialAgentList</a>                        | 53 |
| 7.14.4.12 | <a href="#">neighborsFace</a>                              | 53 |
| 7.14.4.13 | <a href="#">neighborsSite</a>                              | 53 |
| 7.14.4.14 | <a href="#">potential</a>                                  | 53 |
| 7.14.4.15 | <a href="#">registerAgent</a>                              | 54 |
| 7.14.4.16 | <a href="#">registerDefect</a>                             | 54 |
| 7.14.4.17 | <a href="#">registerSpecialAgent</a>                       | 54 |
| 7.14.4.18 | <a href="#">setPotential</a>                               | 54 |
| 7.14.4.19 | <a href="#">sliceIndex</a>                                 | 54 |
| 7.14.4.20 | <a href="#">specialAgentCount</a>                          | 55 |

|   |    |
|---|----|
| 7.14.4.21 toQString . . . . .                           | 55 |
| 7.14.4.22 totalDistance . . . . .                       | 55 |
| 7.14.4.23 unregisterAgent . . . . .                     | 55 |
| 7.14.4.24 unregisterDefect . . . . .                    | 55 |
| 7.14.4.25 unregisterSpecialAgent . . . . .              | 55 |
| 7.14.4.26 volume . . . . .                              | 55 |
| 7.14.4.27 xDistance . . . . .                           | 56 |
| 7.14.4.28 xDistancei . . . . .                          | 56 |
| 7.14.4.29 xImageDistance . . . . .                      | 56 |
| 7.14.4.30 xImageDistancei . . . . .                     | 56 |
| 7.14.4.31 xSize . . . . .                               | 56 |
| 7.14.4.32 xyPlaneArea . . . . .                         | 56 |
| 7.14.4.33 yDistance . . . . .                           | 56 |
| 7.14.4.34 yDistancei . . . . .                          | 57 |
| 7.14.4.35 yImageDistance . . . . .                      | 57 |
| 7.14.4.36 yImageDistancei . . . . .                     | 57 |
| 7.14.4.37 ySize . . . . .                               | 57 |
| 7.14.4.38 zDistance . . . . .                           | 57 |
| 7.14.4.39 zDistancei . . . . .                          | 58 |
| 7.14.4.40 zImageDistance . . . . .                      | 58 |
| 7.14.4.41 zImageDistancei . . . . .                     | 58 |
| 7.14.4.42 zSize . . . . .                               | 58 |
| 7.14.5 Member Data Documentation . . . . .              | 58 |
| 7.14.5.1 m_agents . . . . .                             | 58 |
| 7.14.5.2 m_agentType . . . . .                          | 58 |
| 7.14.5.3 m_potentials . . . . .                         | 59 |
| 7.14.5.4 m_specialAgentCount . . . . .                  | 59 |
| 7.14.5.5 m_specialAgentReserve . . . . .                | 59 |
| 7.14.5.6 m_specialAgents . . . . .                      | 59 |
| 7.14.5.7 m_volume . . . . .                             | 59 |
| 7.14.5.8 m_world . . . . .                              | 59 |
| 7.14.5.9 m_xSize . . . . .                              | 59 |
| 7.14.5.10 m_xyPlaneArea . . . . .                       | 59 |
| 7.14.5.11 m_xzPlaneArea . . . . .                       | 59 |
| 7.14.5.12 m_ySize . . . . .                             | 59 |
| 7.14.5.13 m_yzPlaneArea . . . . .                       | 60 |
| 7.14.5.14 m_zSize . . . . .                             | 60 |
| 7.15 Langmuir::GridImage Class Reference . . . . .      | 60 |
| 7.15.1 Detailed Description . . . . .                   | 60 |
| 7.15.2 Constructor & Destructor Documentation . . . . . | 60 |

|          |   |    |
|----------|---|----|
| 7.15.2.1 | GridImage                                 | 60 |
| 7.15.3   | Member Function Documentation             | 61 |
| 7.15.3.1 | drawCharges                               | 61 |
| 7.15.3.2 | drawSites                                 | 61 |
| 7.15.3.3 | save                                      | 61 |
| 7.15.4   | Member Data Documentation                 | 61 |
| 7.15.4.1 | m_image                                   | 61 |
| 7.15.4.2 | mPainter                                  | 61 |
| 7.15.4.3 | m_world                                   | 61 |
| 7.16     | Langmuir::HoleAgent Class Reference       | 62 |
| 7.16.1   | Detailed Description                      | 62 |
| 7.16.2   | Constructor & Destructor Documentation    | 62 |
| 7.16.2.1 | HoleAgent                                 | 62 |
| 7.16.3   | Member Function Documentation             | 62 |
| 7.16.3.1 | bindingPotential                          | 62 |
| 7.16.3.2 | otherGrid                                 | 63 |
| 7.16.3.3 | otherType                                 | 63 |
| 7.17     | Langmuir::HoleDrainAgent Class Reference  | 63 |
| 7.17.1   | Detailed Description                      | 63 |
| 7.17.2   | Constructor & Destructor Documentation    | 63 |
| 7.17.2.1 | HoleDrainAgent                            | 63 |
| 7.17.2.2 | HoleDrainAgent                            | 64 |
| 7.17.3   | Member Function Documentation             | 64 |
| 7.17.3.1 | energyChange                              | 64 |
| 7.18     | Langmuir::HoleSourceAgent Class Reference | 64 |
| 7.18.1   | Detailed Description                      | 64 |
| 7.18.2   | Constructor & Destructor Documentation    | 64 |
| 7.18.2.1 | HoleSourceAgent                           | 64 |
| 7.18.2.2 | HoleSourceAgent                           | 65 |
| 7.18.3   | Member Function Documentation             | 65 |
| 7.18.3.1 | energyChange                              | 65 |
| 7.18.3.2 | inject                                    | 65 |
| 7.18.3.3 | validToInject                             | 65 |
| 7.19     | Langmuir::KeyValueParser Class Reference  | 65 |
| 7.19.1   | Detailed Description                      | 66 |
| 7.19.2   | Constructor & Destructor Documentation    | 66 |
| 7.19.2.1 | KeyValueParser                            | 66 |
| 7.19.2.2 | ~KeyValueParser                           | 67 |
| 7.19.3   | Member Function Documentation             | 67 |
| 7.19.3.1 | getVariable                               | 67 |

|           |  |    |
|-----------|--|----|
| 7.19.3.2  | parameters                                 | 67 |
| 7.19.3.3  | parse                                      | 67 |
| 7.19.3.4  | registerVariable                           | 67 |
| 7.19.3.5  | save                                       | 67 |
| 7.19.4    | Friends And Related Function Documentation | 67 |
| 7.19.4.1  | operator<<                                 | 67 |
| 7.19.5    | Member Data Documentation                  | 67 |
| 7.19.5.1  | m_orderedNames                             | 67 |
| 7.19.5.2  | m_parameters                               | 67 |
| 7.19.5.3  | m_variableMap                              | 67 |
| 7.19.5.4  | m_world                                    | 68 |
| 7.20      | Langmuir::Logger Class Reference           | 68 |
| 7.20.1    | Detailed Description                       | 69 |
| 7.20.2    | Constructor & Destructor Documentation     | 69 |
| 7.20.2.1  | Logger                                     | 69 |
| 7.20.3    | Member Function Documentation              | 69 |
| 7.20.3.1  | initialize                                 | 69 |
| 7.20.3.2  | reportCarrier                              | 69 |
| 7.20.3.3  | reportExciton                              | 69 |
| 7.20.3.4  | reportFluxStream                           | 69 |
| 7.20.3.5  | reportXYZStream                            | 69 |
| 7.20.3.6  | saveCarriersImage                          | 70 |
| 7.20.3.7  | saveCoulombEnergy                          | 70 |
| 7.20.3.8  | saveDefectImage                            | 70 |
| 7.20.3.9  | saveElectronImage                          | 70 |
| 7.20.3.10 | saveGridPotential                          | 70 |
| 7.20.3.11 | saveHoleImage                              | 70 |
| 7.20.3.12 | saveImage                                  | 70 |
| 7.20.3.13 | saveTrapImage                              | 70 |
| 7.20.4    | Member Data Documentation                  | 70 |
| 7.20.4.1  | m_carrierWriter                            | 70 |
| 7.20.4.2  | m_excitonWriter                            | 70 |
| 7.20.4.3  | m_fluxWriter                               | 70 |
| 7.20.4.4  | m_world                                    | 71 |
| 7.20.4.5  | m_xyzWriter                                | 71 |
| 7.21      | Langmuir::OpenCIHelper Class Reference     | 71 |
| 7.21.1    | Detailed Description                       | 72 |
| 7.21.2    | Constructor & Destructor Documentation     | 72 |
| 7.21.2.1  | OpenCIHelper                               | 72 |
| 7.21.3    | Member Function Documentation              | 72 |

|           |  |    |
|-----------|--|----|
| 7.21.3.1  | <a href="#">compareHostAndDeviceForAllCarriers</a>         | 72 |
| 7.21.3.2  | <a href="#">copySiteAndChargeToHostVector</a>              | 72 |
| 7.21.3.3  | <a href="#">getOutputHost</a>                              | 72 |
| 7.21.3.4  | <a href="#">getOutputHostFuture</a>                        | 72 |
| 7.21.3.5  | <a href="#">initializeOpenCL</a>                           | 73 |
| 7.21.3.6  | <a href="#">launchCoulombKernel1</a>                       | 73 |
| 7.21.3.7  | <a href="#">launchCoulombKernel2</a>                       | 73 |
| 7.21.3.8  | <a href="#">launchGaussKernel1</a>                         | 73 |
| 7.21.3.9  | <a href="#">launchGaussKernel2</a>                         | 73 |
| 7.21.3.10 | <a href="#">toggleOpenCL</a>                               | 73 |
| 7.21.4    | <a href="#">Member Data Documentation</a>                  | 73 |
| 7.21.4.1  | <a href="#">m_world</a>                                    | 73 |
| 7.22      | <a href="#">Langmuir::OutputInfo Class Reference</a>       | 73 |
| 7.22.1    | <a href="#">Detailed Description</a>                       | 74 |
| 7.22.2    | <a href="#">Constructor &amp; Destructor Documentation</a> | 74 |
| 7.22.2.1  | <a href="#">OutputInfo</a>                                 | 74 |
| 7.23      | <a href="#">Langmuir::OutputStream Class Reference</a>     | 74 |
| 7.23.1    | <a href="#">Detailed Description</a>                       | 75 |
| 7.23.2    | <a href="#">Constructor &amp; Destructor Documentation</a> | 75 |
| 7.23.2.1  | <a href="#">OutputStream</a>                               | 75 |
| 7.23.2.2  | <a href="#">~OutputStream</a>                              | 75 |
| 7.23.3    | <a href="#">Member Function Documentation</a>              | 75 |
| 7.23.3.1  | <a href="#">file</a>                                       | 75 |
| 7.23.3.2  | <a href="#">info</a>                                       | 75 |
| 7.23.4    | <a href="#">Member Data Documentation</a>                  | 75 |
| 7.23.4.1  | <a href="#">m_file</a>                                     | 75 |
| 7.23.4.2  | <a href="#">m_info</a>                                     | 75 |
| 7.24      | <a href="#">Langmuir::PBSGPUParser Class Reference</a>     | 76 |
| 7.24.1    | <a href="#">Detailed Description</a>                       | 76 |
| 7.24.2    | <a href="#">Constructor &amp; Destructor Documentation</a> | 76 |
| 7.24.2.1  | <a href="#">PBSGPUParser</a>                               | 76 |
| 7.24.3    | <a href="#">Member Function Documentation</a>              | 77 |
| 7.24.3.1  | <a href="#">aquirePath</a>                                 | 77 |
| 7.24.3.2  | <a href="#">gpu</a>  | 77 |
| 7.24.3.3  | <a href="#">gpus</a>                                       | 77 |
| 7.24.3.4  | <a href="#">node</a>                                       | 77 |
| 7.24.3.5  | <a href="#">nodes</a>                                      | 77 |
| 7.24.3.6  | <a href="#">parse</a>                                      | 77 |
| 7.24.3.7  | <a href="#">setDefault</a>                                 | 77 |
| 7.24.3.8  | <a href="#">setPath</a>                                    | 77 |

|           |  |    |
|-----------|--|----|
| 7.24.3.9  | size                                   | 77 |
| 7.24.4    | Member Data Documentation              | 77 |
| 7.24.4.1  | m_gpus                                 | 77 |
| 7.24.4.2  | m_nodes                                | 77 |
| 7.24.4.3  | m_path                                 | 78 |
| 7.25      | Langmuir::Potential Class Reference    | 78 |
| 7.25.1    | Detailed Description                   | 79 |
| 7.25.2    | Constructor & Destructor Documentation | 79 |
| 7.25.2.1  | Potential                              | 79 |
| 7.25.3    | Member Function Documentation          | 79 |
| 7.25.3.1  | coulombD                               | 79 |
| 7.25.3.2  | coulombE                               | 79 |
| 7.25.3.3  | coulombH                               | 79 |
| 7.25.3.4  | coulombImageD                          | 80 |
| 7.25.3.5  | coulombImageE                          | 80 |
| 7.25.3.6  | coulombImageH                          | 80 |
| 7.25.3.7  | gaussD                                 | 80 |
| 7.25.3.8  | gaussE                                 | 80 |
| 7.25.3.9  | gaussH                                 | 80 |
| 7.25.3.10 | gaussImageD                            | 80 |
| 7.25.3.11 | gaussImageE                            | 81 |
| 7.25.3.12 | gaussImageH                            | 81 |
| 7.25.3.13 | precalculateArrays                     | 81 |
| 7.25.3.14 | setPotentialGate                       | 81 |
| 7.25.3.15 | setPotentialLinear                     | 81 |
| 7.25.3.16 | setPotentialTraps                      | 81 |
| 7.25.3.17 | setPotentialZero                       | 81 |
| 7.25.3.18 | updateCouplingConstants                | 81 |
| 7.25.4    | Member Data Documentation              | 82 |
| 7.25.4.1  | m_world                                | 82 |
| 7.26      | Langmuir::Random Class Reference       | 82 |
| 7.26.1    | Detailed Description                   | 83 |
| 7.26.2    | Constructor & Destructor Documentation | 83 |
| 7.26.2.1  | Random                                 | 83 |
| 7.26.2.2  | ~Random                                | 83 |
| 7.26.3    | Member Function Documentation          | 83 |
| 7.26.3.1  | chooseNo                               | 83 |
| 7.26.3.2  | chooseYes                              | 83 |
| 7.26.3.3  | integer                                | 83 |
| 7.26.3.4  | metropolis                             | 84 |

|           |  |    |
|-----------|--|----|
| 7.26.3.5  | metropolisWithCoupling                       | 84 |
| 7.26.3.6  | normal                                       | 84 |
| 7.26.3.7  | random                                       | 84 |
| 7.26.3.8  | range  | 84 |
| 7.26.3.9  | seed   | 84 |
| 7.26.3.10 | seed   | 84 |
| 7.26.4    | Friends And Related Function Documentation   | 84 |
| 7.26.4.1  | operator<<                                   | 85 |
| 7.26.4.2  | operator<<                                   | 85 |
| 7.26.4.3  | operator<<                                   | 85 |
| 7.26.4.4  | operator>>                                   | 85 |
| 7.26.4.5  | operator>>                                   | 85 |
| 7.26.4.6  | operator>>                                   | 85 |
| 7.26.5    | Member Data Documentation                    | 85 |
| 7.26.5.1  | generator01                                  | 85 |
| 7.26.5.2  | m_seed                                       | 85 |
| 7.26.5.3  | twister                                      | 86 |
| 7.27      | Langmuir::RecombinationAgent Class Reference | 86 |
| 7.27.1    | Detailed Description                         | 86 |
| 7.27.2    | Constructor & Destructor Documentation       | 86 |
| 7.27.2.1  | RecombinationAgent                           | 86 |
| 7.27.3    | Member Function Documentation                | 86 |
| 7.27.3.1  | energyChange                                 | 86 |
| 7.27.3.2  | guessProbability                             | 87 |
| 7.27.3.3  | tryToAccept                                  | 87 |
| 7.28      | Langmuir::Simulation Class Reference         | 87 |
| 7.28.1    | Detailed Description                         | 88 |
| 7.28.2    | Constructor & Destructor Documentation       | 88 |
| 7.28.2.1  | Simulation                                   | 88 |
| 7.28.2.2  | ~Simulation                                  | 88 |
| 7.28.3    | Member Function Documentation                | 88 |
| 7.28.3.1  | balanceCharges                               | 88 |
| 7.28.3.2  | chargeAgentCoulombInteractionQtConcurrentCPU | 88 |
| 7.28.3.3  | chargeAgentCoulombInteractionQtConcurrentGPU | 88 |
| 7.28.3.4  | nextTick                                     | 88 |
| 7.28.3.5  | performInjections                            | 88 |
| 7.28.3.6  | performIterations                            | 88 |
| 7.28.3.7  | performRecombinations                        | 89 |
| 7.28.4    | Member Data Documentation                    | 89 |
| 7.28.4.1  | m_world                                      | 89 |

|           |   |    |
|-----------|---|----|
| 7.29      | Langmuir::SimulationParameters Struct Reference | 89 |
| 7.29.1    | Detailed Description                            | 92 |
| 7.29.2    | Constructor & Destructor Documentation          | 93 |
| 7.29.2.1  | SimulationParameters                            | 93 |
| 7.29.3    | Member Data Documentation                       | 93 |
| 7.29.3.1  | balanceCharges                                  | 93 |
| 7.29.3.2  | boltzmannConstant                               | 93 |
| 7.29.3.3  | coulombCarriers                                 | 93 |
| 7.29.3.4  | coulombGaussianSigma                            | 93 |
| 7.29.3.5  | currentStep                                     | 93 |
| 7.29.3.6  | defectPercentage                                | 93 |
| 7.29.3.7  | defectsCharge                                   | 93 |
| 7.29.3.8  | dielectricConstant                              | 93 |
| 7.29.3.9  | drainRate                                       | 93 |
| 7.29.3.10 | eDrainLRate                                     | 93 |
| 7.29.3.11 | eDrainRRate                                     | 93 |
| 7.29.3.12 | electronPercentage                              | 94 |
| 7.29.3.13 | electrostaticCutoff                             | 94 |
| 7.29.3.14 | electrostaticPrefactor                          | 94 |
| 7.29.3.15 | elementaryCharge                                | 94 |
| 7.29.3.16 | eSourceLRate                                    | 94 |
| 7.29.3.17 | eSourceRRate                                    | 94 |
| 7.29.3.18 | excitonBinding                                  | 94 |
| 7.29.3.19 | gaussianStdev                                   | 94 |
| 7.29.3.20 | generationRate                                  | 94 |
| 7.29.3.21 | gridFactor                                      | 94 |
| 7.29.3.22 | gridX   | 94 |
| 7.29.3.23 | gridY   | 94 |
| 7.29.3.24 | gridZ   | 95 |
| 7.29.3.25 | hDrainLRate                                     | 95 |
| 7.29.3.26 | hDrainRRate                                     | 95 |
| 7.29.3.27 | holePercentage                                  | 95 |
| 7.29.3.28 | hoppingRange                                    | 95 |
| 7.29.3.29 | hSourceLRate                                    | 95 |
| 7.29.3.30 | hSourceRRate                                    | 95 |
| 7.29.3.31 | imageCarriers                                   | 95 |
| 7.29.3.32 | imageDefects                                    | 95 |
| 7.29.3.33 | imageTraps                                      | 95 |
| 7.29.3.34 | inverseKT                                       | 95 |
| 7.29.3.35 | iterationsPrint                                 | 95 |



|  |    |
|--|----|
| 7.29.3.36 iterationsReal . . . . .         | 96 |
| 7.29.3.37 maxThreads . . . . .             | 96 |
| 7.29.3.38 okCL . . . . .                   | 96 |
| 7.29.3.39 opencldDeviceID . . . . .        | 96 |
| 7.29.3.40 opencldThreshold . . . . .       | 96 |
| 7.29.3.41 outputChkTrapPotential . . . . . | 96 |
| 7.29.3.42 outputCoulomb . . . . .          | 96 |
| 7.29.3.43 outputIdsOnDelete . . . . .      | 96 |
| 7.29.3.44 outputIdsOnEncounter . . . . .   | 96 |
| 7.29.3.45 outputIsOn . . . . .             | 96 |
| 7.29.3.46 outputPotential . . . . .        | 96 |
| 7.29.3.47 outputPrecision . . . . .        | 96 |
| 7.29.3.48 outputStepChk . . . . .          | 97 |
| 7.29.3.49 outputStub . . . . .             | 97 |
| 7.29.3.50 outputWidth . . . . .            | 97 |
| 7.29.3.51 outputXyz . . . . .              | 97 |
| 7.29.3.52 outputXyzD . . . . .             | 97 |
| 7.29.3.53 outputXyzE . . . . .             | 97 |
| 7.29.3.54 outputXyzH . . . . .             | 97 |
| 7.29.3.55 outputXyzMode . . . . .          | 97 |
| 7.29.3.56 outputXyzT . . . . .             | 97 |
| 7.29.3.57 permittivitySpace . . . . .      | 97 |
| 7.29.3.58 randomSeed . . . . .             | 97 |
| 7.29.3.59 recombinationRange . . . . .     | 97 |
| 7.29.3.60 recombinationRate . . . . .      | 98 |
| 7.29.3.61 seedCharges . . . . .            | 98 |
| 7.29.3.62 seedPercentage . . . . .         | 98 |
| 7.29.3.63 simulationStart . . . . .        | 98 |
| 7.29.3.64 simulationType . . . . .         | 98 |
| 7.29.3.65 slopeZ . . . . .                 | 98 |
| 7.29.3.66 sourceCoulomb . . . . .          | 98 |
| 7.29.3.67 sourceMetropolis . . . . .       | 98 |
| 7.29.3.68 sourceRate . . . . .             | 98 |
| 7.29.3.69 sourceScaleArea . . . . .        | 98 |
| 7.29.3.70 temperatureKelvin . . . . .      | 98 |
| 7.29.3.71 trapPercentage . . . . .         | 98 |
| 7.29.3.72 trapPotential . . . . .          | 99 |
| 7.29.3.73 useOpenCL . . . . .              | 99 |
| 7.29.3.74 voltageLeft . . . . .            | 99 |
| 7.29.3.75 voltageRight . . . . .           | 99 |

|           |   |     |
|-----------|---|-----|
| 7.29.3.76 | workSize  | 99  |
| 7.29.3.77 | workX   | 99  |
| 7.29.3.78 | workY   | 99  |
| 7.29.3.79 | workZ   | 99  |
| 7.30      | Langmuir::SourceAgent Class Reference                 | 99  |
| 7.30.1    | Detailed Description                                  | 100 |
| 7.30.2    | Constructor & Destructor Documentation                | 100 |
| 7.30.2.1  | SourceAgent   | 100 |
| 7.30.3    | Member Function Documentation                         | 100 |
| 7.30.3.1  | chooseSite  | 100 |
| 7.30.3.2  | inject  | 100 |
| 7.30.3.3  | randomNeighborSiteID                                  | 101 |
| 7.30.3.4  | randomSiteID  | 101 |
| 7.30.3.5  | shouldTransport                                       | 101 |
| 7.30.3.6  | tryToInject   | 101 |
| 7.30.3.7  | tryToSeed   | 101 |
| 7.30.3.8  | tryToSeed   | 101 |
| 7.30.3.9  | validToInject   | 102 |
| 7.31      | Langmuir::Tolerance Class Reference                   | 102 |
| 7.31.1    | Detailed Description                                  | 103 |
| 7.31.2    | Constructor & Destructor Documentation                | 103 |
| 7.31.2.1  | Tolerance   | 103 |
| 7.31.3    | Member Function Documentation                         | 103 |
| 7.31.3.1  | check   | 103 |
| 7.31.3.2  | checkCriteria   | 103 |
| 7.31.3.3  | converged   | 103 |
| 7.31.3.4  | percentChange   | 103 |
| 7.31.3.5  | setCurrent  | 103 |
| 7.31.3.6  | setPrevious   | 103 |
| 7.31.4    | Member Data Documentation                             | 103 |
| 7.31.4.1  | m_converged   | 104 |
| 7.31.4.2  | m_criteria  | 104 |
| 7.31.4.3  | m_current   | 104 |
| 7.31.4.4  | m_flux  | 104 |
| 7.31.4.5  | m_previous  | 104 |
| 7.31.4.6  | m_world   | 104 |
| 7.32      | Langmuir::TypedVariable< T > Class Template Reference | 104 |
| 7.32.1    | Detailed Description                                  | 105 |
| 7.32.2    | Constructor & Destructor Documentation                | 106 |
| 7.32.2.1  | TypedVariable   | 106 |

|           |  |     |
|-----------|--|-----|
| 7.32.3    | Member Function Documentation              | 106 |
| 7.32.3.1  | convert                                    | 106 |
| 7.32.3.2  | convert                                    | 106 |
| 7.32.3.3  | convert                                    | 106 |
| 7.32.3.4  | convert                                    | 106 |
| 7.32.3.5  | convert                                    | 106 |
| 7.32.3.6  | convert                                    | 106 |
| 7.32.3.7  | convert                                    | 106 |
| 7.32.3.8  | convert                                    | 106 |
| 7.32.3.9  | convert                                    | 106 |
| 7.32.3.10 | convert                                    | 106 |
| 7.32.3.11 | key  | 107 |
| 7.32.3.12 | keyValue                                   | 107 |
| 7.32.3.13 | keyValue                                   | 107 |
| 7.32.3.14 | read                                       | 107 |
| 7.32.3.15 | value                                      | 107 |
| 7.32.3.16 | value                                      | 107 |
| 7.32.3.17 | value                                      | 107 |
| 7.32.3.18 | value                                      | 107 |
| 7.32.3.19 | write                                      | 108 |
| 7.32.4    | Member Data Documentation                  | 108 |
| 7.32.4.1  | m_value                                    | 108 |
| 7.33      | Langmuir::Variable Class Reference         | 108 |
| 7.33.1    | Detailed Description                       | 109 |
| 7.33.2    | Member Enumeration Documentation           | 109 |
| 7.33.2.1  | VariableModeFlag                           | 109 |
| 7.33.3    | Constructor & Destructor Documentation     | 109 |
| 7.33.3.1  | Variable                                   | 109 |
| 7.33.4    | Member Function Documentation              | 109 |
| 7.33.4.1  | isConstant                                 | 109 |
| 7.33.4.2  | key  | 109 |
| 7.33.4.3  | keyValue                                   | 110 |
| 7.33.4.4  | mode                                       | 110 |
| 7.33.4.5  | read                                       | 110 |
| 7.33.4.6  | value                                      | 110 |
| 7.33.4.7  | write                                      | 110 |
| 7.33.5    | Friends And Related Function Documentation | 110 |
| 7.33.5.1  | operator<<                                 | 110 |
| 7.33.5.2  | operator<<                                 | 110 |
| 7.33.5.3  | operator<<                                 | 110 |

|           |  |     |
|-----------|--|-----|
| 7.33.6    | Member Data Documentation . . . . .              | 110 |
| 7.33.6.1  | m_key . . . . .                                  | 110 |
| 7.33.6.2  | m_mode . . . . .                                 | 111 |
| 7.34      | Langmuir::World Class Reference . . . . .        | 111 |
| 7.34.1    | Detailed Description . . . . .                   | 116 |
| 7.34.2    | Constructor & Destructor Documentation . . . . . | 116 |
| 7.34.2.1  | World . . . . .                                  | 116 |
| 7.34.2.2  | ~World . . . . .                                 | 116 |
| 7.34.3    | Member Function Documentation . . . . .          | 116 |
| 7.34.3.1  | atMaxCharges . . . . .                           | 116 |
| 7.34.3.2  | atMaxElectrons . . . . .                         | 116 |
| 7.34.3.3  | atMaxHoles . . . . .                             | 116 |
| 7.34.3.4  | chargesAreBalanced . . . . .                     | 116 |
| 7.34.3.5  | checkPointer . . . . .                           | 116 |
| 7.34.3.6  | couplingConstants . . . . .                      | 116 |
| 7.34.3.7  | createDrains . . . . .                           | 116 |
| 7.34.3.8  | createSources . . . . .                          | 117 |
| 7.34.3.9  | defectSiteIDs . . . . .                          | 117 |
| 7.34.3.10 | drains . . . . .                                 | 117 |
| 7.34.3.11 | eDrains . . . . .                                | 117 |
| 7.34.3.12 | electronDrainAgentLeft . . . . .                 | 117 |
| 7.34.3.13 | electronDrainAgentRight . . . . .                | 117 |
| 7.34.3.14 | electronGrid . . . . .                           | 117 |
| 7.34.3.15 | electrons . . . . .                              | 117 |
| 7.34.3.16 | electronsMinusHoles . . . . .                    | 117 |
| 7.34.3.17 | electronSourceAgentLeft . . . . .                | 117 |
| 7.34.3.18 | electronSourceAgentRight . . . . .               | 117 |
| 7.34.3.19 | eR . . . . .                                     | 117 |
| 7.34.3.20 | eSources . . . . .                               | 118 |
| 7.34.3.21 | excitonSourceAgent . . . . .                     | 118 |
| 7.34.3.22 | fluxes . . . . .                                 | 118 |
| 7.34.3.23 | hDrains . . . . .                                | 118 |
| 7.34.3.24 | holeDrainAgentLeft . . . . .                     | 118 |
| 7.34.3.25 | holeDrainAgentRight . . . . .                    | 118 |
| 7.34.3.26 | holeGrid . . . . .                               | 118 |
| 7.34.3.27 | holes . . . . .                                  | 118 |
| 7.34.3.28 | holesMinusElectrons . . . . .                    | 118 |
| 7.34.3.29 | holeSourceAgentLeft . . . . .                    | 118 |
| 7.34.3.30 | holeSourceAgentRight . . . . .                   | 118 |
| 7.34.3.31 | hSources . . . . .                               | 118 |

|  |     |
|--|-----|
| 7.34.3.32 initialize                       | 119 |
| 7.34.3.33 iR                               | 119 |
| 7.34.3.34 keyValueParser                   | 119 |
| 7.34.3.35 logger                           | 119 |
| 7.34.3.36 maxChargeAgents                  | 119 |
| 7.34.3.37 maxChargeAgentsAndChargedDefects | 119 |
| 7.34.3.38 maxDefects                       | 119 |
| 7.34.3.39 maxElectronAgents                | 119 |
| 7.34.3.40 maxHoleAgents                    | 119 |
| 7.34.3.41 maxTraps                         | 119 |
| 7.34.3.42 numChargeAgents                  | 119 |
| 7.34.3.43 numChargeAgentsAndChargedDefects | 120 |
| 7.34.3.44 numDefects                       | 120 |
| 7.34.3.45 numElectronAgents                | 120 |
| 7.34.3.46 numHoleAgents                    | 120 |
| 7.34.3.47 numTraps                         | 120 |
| 7.34.3.48 opencl                           | 120 |
| 7.34.3.49 parameters                       | 120 |
| 7.34.3.50 percentElectronAgents            | 120 |
| 7.34.3.51 percentHoleAgents                | 120 |
| 7.34.3.52 placeDefects                     | 120 |
| 7.34.3.53 placeElectrons                   | 120 |
| 7.34.3.54 placeHoles                       | 121 |
| 7.34.3.55 potential                        | 121 |
| 7.34.3.56 R1                               | 121 |
| 7.34.3.57 R2                               | 121 |
| 7.34.3.58 randomNumberGenerator            | 121 |
| 7.34.3.59 reachedChargeAgents              | 121 |
| 7.34.3.60 reachedElectronAgents            | 121 |
| 7.34.3.61 reachedHoleAgents                | 121 |
| 7.34.3.62 recombinationAgent               | 121 |
| 7.34.3.63 setFluxInfo                      | 122 |
| 7.34.3.64 sl                               | 122 |
| 7.34.3.65 sources                          | 122 |
| 7.34.3.66 trapSiteIDs                      | 122 |
| 7.34.3.67 trapSitePotentials               | 122 |
| 7.34.3.68 xDrains                          | 122 |
| 7.34.3.69 xSources                         | 122 |
| 7.34.4 Member Data Documentation           | 122 |
| 7.34.4.1 m_checkPointer                    | 122 |

|           |  |     |
|-----------|--|-----|
| 7.34.4.2  | <a href="#">m_couplingConstants</a>        | 122 |
| 7.34.4.3  | <a href="#">m_defectSiteIDs</a>            | 122 |
| 7.34.4.4  | <a href="#">m_drains</a>                   | 122 |
| 7.34.4.5  | <a href="#">m_eDrains</a>                  | 123 |
| 7.34.4.6  | <a href="#">m_electronDrainAgentLeft</a>   | 123 |
| 7.34.4.7  | <a href="#">m_electronDrainAgentRight</a>  | 123 |
| 7.34.4.8  | <a href="#">m_electronGrid</a>             | 123 |
| 7.34.4.9  | <a href="#">m_electrons</a>                | 123 |
| 7.34.4.10 | <a href="#">m_electronSourceAgentLeft</a>  | 123 |
| 7.34.4.11 | <a href="#">m_electronSourceAgentRight</a> | 123 |
| 7.34.4.12 | <a href="#">m_eR</a>                       | 123 |
| 7.34.4.13 | <a href="#">m_eSources</a>                 | 123 |
| 7.34.4.14 | <a href="#">m_excitonSourceAgent</a>       | 123 |
| 7.34.4.15 | <a href="#">m_fluxAgents</a>               | 123 |
| 7.34.4.16 | <a href="#">m_hDrains</a>                  | 123 |
| 7.34.4.17 | <a href="#">m_holeDrainAgentLeft</a>       | 124 |
| 7.34.4.18 | <a href="#">m_holeDrainAgentRight</a>      | 124 |
| 7.34.4.19 | <a href="#">m_holeGrid</a>                 | 124 |
| 7.34.4.20 | <a href="#">m_holes</a>                    | 124 |
| 7.34.4.21 | <a href="#">m_holeSourceAgentLeft</a>      | 124 |
| 7.34.4.22 | <a href="#">m_holeSourceAgentRight</a>     | 124 |
| 7.34.4.23 | <a href="#">m_hSources</a>                 | 124 |
| 7.34.4.24 | <a href="#">m_iR</a>                       | 124 |
| 7.34.4.25 | <a href="#">m_keyValueParser</a>           | 124 |
| 7.34.4.26 | <a href="#">m_logger</a>                   | 124 |
| 7.34.4.27 | <a href="#">m_maxDefects</a>               | 124 |
| 7.34.4.28 | <a href="#">m_maxElectrons</a>             | 124 |
| 7.34.4.29 | <a href="#">m_maxHoles</a>                 | 125 |
| 7.34.4.30 | <a href="#">m_maxTraps</a>                 | 125 |
| 7.34.4.31 | <a href="#">m_ocl</a>                      | 125 |
| 7.34.4.32 | <a href="#">m_parameters</a>               | 125 |
| 7.34.4.33 | <a href="#">m_potential</a>                | 125 |
| 7.34.4.34 | <a href="#">m_R1</a>                       | 125 |
| 7.34.4.35 | <a href="#">m_R2</a>                       | 125 |
| 7.34.4.36 | <a href="#">m_rand</a>                     | 125 |
| 7.34.4.37 | <a href="#">m_recombinationAgent</a>       | 125 |
| 7.34.4.38 | <a href="#">m_sl</a>                       | 125 |
| 7.34.4.39 | <a href="#">m_sources</a>                  | 125 |
| 7.34.4.40 | <a href="#">m_trapSiteIDs</a>              | 126 |
| 7.34.4.41 | <a href="#">m_trapSitePotentials</a>       | 126 |

|           |  |            |
|-----------|--|------------|
| 7.34.4.42 | <a href="#">m_xDrains</a>                                  | 126        |
| 7.34.4.43 | <a href="#">m_xSources</a>                                 | 126        |
| 7.35      | <a href="#">Langmuir::XYZWriter Class Reference</a>        | 126        |
| 7.35.1    | <a href="#">Detailed Description</a>                       | 126        |
| 7.35.2    | <a href="#">Constructor &amp; Destructor Documentation</a> | 127        |
| 7.35.2.1  | <a href="#">XYZWriter</a>                                  | 127        |
| 7.35.3    | <a href="#">Member Function Documentation</a>              | 127        |
| 7.35.3.1  | <a href="#">write</a>                                      | 127        |
| 7.35.3.2  | <a href="#">writeVMDInitFile</a>                           | 127        |
| 7.35.4    | <a href="#">Member Data Documentation</a>                  | 127        |
| 7.35.4.1  | <a href="#">m_stream</a>                                   | 127        |
| 7.35.4.2  | <a href="#">m_world</a>                                    | 127        |
| <b>8</b>  | <b><a href="#">File Documentation</a></b>                  | <b>129</b> |
| 8.1       | <a href="#">agent.h File Reference</a>                     | 129        |
| 8.2       | <a href="#">chargeagent.cpp File Reference</a>             | 129        |
| 8.3       | <a href="#">chargeagent.h File Reference</a>               | 130        |
| 8.4       | <a href="#">checkpointer.cpp File Reference</a>            | 130        |
| 8.5       | <a href="#">checkpointer.h File Reference</a>              | 131        |
| 8.6       | <a href="#">copy_to_frank.py File Reference</a>            | 131        |
| 8.7       | <a href="#">cubicgrid.cpp File Reference</a>               | 131        |
| 8.8       | <a href="#">cubicgrid.h File Reference</a>                 | 132        |
| 8.9       | <a href="#">drainagent.cpp File Reference</a>              | 132        |
| 8.10      | <a href="#">drainagent.h File Reference</a>                | 133        |
| 8.11      | <a href="#">fluxagent.cpp File Reference</a>               | 133        |
| 8.12      | <a href="#">fluxagent.h File Reference</a>                 | 133        |
| 8.13      | <a href="#">gridview.cpp File Reference</a>                | 134        |
| 8.14      | <a href="#">keyvalueparser.cpp File Reference</a>          | 134        |
| 8.15      | <a href="#">keyvalueparser.h File Reference</a>            | 134        |
| 8.16      | <a href="#">langmuir.cpp File Reference</a>                | 135        |
| 8.16.1    | <a href="#">Function Documentation</a>                     | 135        |
| 8.16.1.1  | <a href="#">alterMaxThreads</a>                            | 135        |
| 8.16.1.2  | <a href="#">main</a>                                       | 135        |
| 8.16.1.3  | <a href="#">progress</a>                                   | 135        |
| 8.17      | <a href="#">langmuirview.cpp File Reference</a>            | 135        |
| 8.17.1    | <a href="#">Function Documentation</a>                     | 135        |
| 8.17.1.1  | <a href="#">main</a>                                       | 135        |
| 8.18      | <a href="#">openclhelper.cpp File Reference</a>            | 136        |
| 8.19      | <a href="#">openclhelper.h File Reference</a>              | 136        |
| 8.19.1    | <a href="#">Macro Definition Documentation</a>             | 136        |

|          |   |     |
|----------|---|-----|
| 8.19.1.1 | <a href="#">__CL_ENABLE_EXCEPTIONS</a>          | 136 |
| 8.20     | <a href="#">output.cpp File Reference</a>       | 136 |
| 8.20.1   | <a href="#">Function Documentation</a>          | 137 |
| 8.20.1.1 | <a href="#">newline</a>                         | 137 |
| 8.20.1.2 | <a href="#">space</a>                           | 137 |
| 8.21     | <a href="#">output.h File Reference</a>         | 137 |
| 8.21.1   | <a href="#">Function Documentation</a>          | 138 |
| 8.21.1.1 | <a href="#">newline</a>                         | 138 |
| 8.21.1.2 | <a href="#">space</a>                           | 138 |
| 8.22     | <a href="#">parameters.h File Reference</a>     | 138 |
| 8.23     | <a href="#">pbsgpuparser.cpp File Reference</a> | 138 |
| 8.24     | <a href="#">pbsgpuparser.h File Reference</a>   | 139 |
| 8.25     | <a href="#">potential.cpp File Reference</a>    | 139 |
| 8.26     | <a href="#">potential.h File Reference</a>      | 139 |
| 8.26.1   | <a href="#">Macro Definition Documentation</a>  | 140 |
| 8.26.1.1 | <a href="#">BOOST_DISABLE_ASSERTS</a>           | 140 |
| 8.27     | <a href="#">rand.cpp File Reference</a>         | 140 |
| 8.28     | <a href="#">rand.h File Reference</a>           | 140 |
| 8.29     | <a href="#">README.md File Reference</a>        | 140 |
| 8.30     | <a href="#">simulation.cpp File Reference</a>   | 141 |
| 8.31     | <a href="#">simulation.h File Reference</a>     | 141 |
| 8.32     | <a href="#">sourceagent.cpp File Reference</a>  | 141 |
| 8.33     | <a href="#">sourceagent.h File Reference</a>    | 142 |
| 8.34     | <a href="#">test.cpp File Reference</a>         | 142 |
| 8.34.1   | <a href="#">Function Documentation</a>          | 142 |
| 8.34.1.1 | <a href="#">main</a>                            | 142 |
| 8.35     | <a href="#">tolerance.cpp File Reference</a>    | 142 |
| 8.36     | <a href="#">tolerance.h File Reference</a>      | 143 |
| 8.37     | <a href="#">variable.h File Reference</a>       | 143 |
| 8.38     | <a href="#">world.cpp File Reference</a>        | 144 |
| 8.39     | <a href="#">world.h File Reference</a>          | 144 |
| 8.39.1   | <a href="#">Macro Definition Documentation</a>  | 144 |
| 8.39.1.1 | <a href="#">BOOST_DISABLE_ASSERTS</a>           | 144 |
| 8.40     | <a href="#">writer.cpp File Reference</a>       | 145 |
| 8.41     | <a href="#">writer.h File Reference</a>         | 145 |



# Chapter 1

## LANGMUIR

- This is the source code for the [Langmuir](#) engine for charge transfer simulations in organic electronics.

### BUILD INSTRUCTIONS

- In order to build the [Langmuir](#) engine the following dependencies are required,
  - Qt4
  - Boost
  - CMake
- The following are optional:
  - OpenCL 1.1
  - OpenGL
  - Doxygen

### SIMPLE BUILD

- mkdir build
- cd build
- cmake ../
- make -j 4
- make install
- make doc

### NOTES

- ON MAC
  - cmake -DCMAKE\_OSX\_ARCHITECTURES=i386 ../
- OPENCL
  - cmake -DOPENCL\_INCLUDE\_DIRS=...
- Documentation
  - open doc/html/index.html in a web browser

## CLANG SCAN-BUILD

- `mkdir build`
- `cd build`
- `scan-build -v cmake ..`
- `scan-build -v -k -analyze-headers -stats -o . make -j 4`
- `scan-view scan-build-output-dir`

## NOTES

- `scan-build-output-dir` will be in the current directory and have the current date for its name

## Chapter 2

# Namespace Index

### 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

|                               |    |
|-------------------------------|----|
| <a href="#">copy_to_frank</a> | 11 |
| <a href="#">Langmuir</a>      | 11 |



## Chapter 3

# Class Index

### 3.1 Class Hierarchy

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

|  |     |
|--|-----|
| Langmuir::Agent . . . . .                | 17  |
| Langmuir::ChargeAgent . . . . .          | 21  |
| Langmuir::ElectronAgent . . . . .        | 33  |
| Langmuir::HoleAgent . . . . .            | 62  |
| Langmuir::FluxAgent . . . . .            | 39  |
| Langmuir::DrainAgent . . . . .           | 33  |
| Langmuir::ElectronDrainAgent . . . . .   | 35  |
| Langmuir::HoleDrainAgent . . . . .       | 63  |
| Langmuir::RecombinationAgent . . . . .   | 86  |
| Langmuir::SourceAgent . . . . .          | 99  |
| Langmuir::ElectronSourceAgent . . . . .  | 36  |
| Langmuir::ExcitonSourceAgent . . . . .   | 37  |
| Langmuir::HoleSourceAgent . . . . .      | 64  |
| Langmuir::CarrierWriter . . . . .        | 20  |
| Langmuir::CheckPointer . . . . .         | 26  |
| Langmuir::ConfigurationInfo . . . . .    | 32  |
| Langmuir::ExcitonWriter . . . . .        | 38  |
| Langmuir::FluxWriter . . . . .           | 46  |
| Langmuir::Grid . . . . .                 | 47  |
| Langmuir::GridImage . . . . .            | 60  |
| Langmuir::KeyValueParser . . . . .       | 65  |
| Langmuir::Logger . . . . .               | 68  |
| Langmuir::OpenCIHelper . . . . .         | 71  |
| Langmuir::OutputInfo . . . . .           | 73  |
| Langmuir::OutputStream . . . . .         | 74  |
| Langmuir::PBSGPUParser . . . . .         | 76  |
| Langmuir::Potential . . . . .            | 78  |
| Langmuir::Random . . . . .               | 82  |
| Langmuir::Simulation . . . . .           | 87  |
| Langmuir::SimulationParameters . . . . . | 89  |
| Langmuir::Tolerance . . . . .            | 102 |
| Langmuir::Variable . . . . .             | 108 |
| Langmuir::TypedVariable< T > . . . . .   | 104 |
| Langmuir::World . . . . .                | 111 |
| Langmuir::XYZWriter . . . . .            | 126 |



## Chapter 4

# Class Index

### 4.1 Class List

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

|   |   |    |
|---|---|----|
| <a href="#">Langmuir::Agent</a>               | A class that abstractly represents an object that can occupy grid sites . . . . .               | 17 |
| <a href="#">Langmuir::CarrierWriter</a>       | A class to output carrier stats (lifetime and pathlength) . . . . .                             | 20 |
| <a href="#">Langmuir::ChargeAgent</a>         | A class to represent moving charged particles . . . . .   | 21 |
| <a href="#">Langmuir::CheckPointer</a>        | A class to read and write checkpoint files . . . . .  | 26 |
| <a href="#">Langmuir::ConfigurationInfo</a>   | A struct to temporarily store site IDs . . . . .  | 32 |
| <a href="#">Langmuir::DrainAgent</a>          | A class to remove charges . . . . .   | 33 |
| <a href="#">Langmuir::ElectronAgent</a>       | A class to represent moving negative charges . . . . .  | 33 |
| <a href="#">Langmuir::ElectronDrainAgent</a>  | A class to remove ElectronAgents . . . . .  | 35 |
| <a href="#">Langmuir::ElectronSourceAgent</a> | A class to inject ElectronAgents . . . . .  | 36 |
| <a href="#">Langmuir::ExcitonSourceAgent</a>  | A class to inject Excitons . . . . .  | 37 |
| <a href="#">Langmuir::ExcitonWriter</a>       | A class to output exciton stats (lifetime and pathlength) . . . . .                             | 38 |
| <a href="#">Langmuir::FluxAgent</a>           | A class to change the number of carriers in the system . . . . .                                | 39 |
| <a href="#">Langmuir::FluxWriter</a>          | A class to output source and drain info . . . . .   | 46 |
| <a href="#">Langmuir::Grid</a>                | A class to hold Agents, calculate their positions, and store the background potential . . . . . | 47 |
| <a href="#">Langmuir::GridImage</a>           | A class to draw images of the grid . . . . .  | 60 |
| <a href="#">Langmuir::HoleAgent</a>           | A class to represent moving positive charges . . . . .  | 62 |
| <a href="#">Langmuir::HoleDrainAgent</a>      | A class to remove HoleAgents . . . . .  | 63 |
| <a href="#">Langmuir::HoleSourceAgent</a>     | A class to inject HoleAgents . . . . .  | 64 |
| <a href="#">Langmuir::KeyValueParser</a>      | A class to read the parameters and store them in the correct place . . . . .                    | 65 |

|  |     |
|--|-----|
| <a href="#">Langmuir::Logger</a>   |     |
| A class that organizes output . . . . .  | 68  |
| <a href="#">Langmuir::OpenCLHelper</a>   |     |
| A Class to run OpenCL calculations . . . . .   | 71  |
| <a href="#">Langmuir::OutputInfo</a>   |     |
| A class to generate file names using the <a href="#">SimulationParameters</a> . . . . .      | 73  |
| <a href="#">Langmuir::OutputStream</a>   |     |
| A class to combine QFile, QTextStream and <a href="#">OutputInfo</a> (QFileInfo) . . . . .   | 74  |
| <a href="#">Langmuir::PBSGPUParser</a> . . . . .   | 76  |
| <a href="#">Langmuir::Potential</a>  |     |
| A class to calculate the potential . . . . .   | 78  |
| <a href="#">Langmuir::Random</a>   |     |
| A class to generate random numbers . . . . .   | 82  |
| <a href="#">Langmuir::RecombinationAgent</a>   |     |
| A class to remove Excitons . . . . .   | 86  |
| <a href="#">Langmuir::Simulation</a>   |     |
| A class to orchestrate the calculation . . . . .   | 87  |
| <a href="#">Langmuir::SimulationParameters</a>   |     |
| A struct to store all simulation options To add new variables, follow these steps: . . . . . | 89  |
| <a href="#">Langmuir::SourceAgent</a>  |     |
| A class to inject charges . . . . .  | 99  |
| <a href="#">Langmuir::Tolerance</a>  |     |
| A class to check if the simulation is converging . . . . .                                   | 102 |
| <a href="#">Langmuir::TypedVariable&lt; T &gt;</a>   |     |
| A template class to map between variable names (keys) and locations (references) . . . . .   | 104 |
| <a href="#">Langmuir::Variable</a>   |     |
| A class to map between variable names (keys) and locations (references) . . . . .            | 108 |
| <a href="#">Langmuir::World</a>  |     |
| A class to hold all objects in a simulation . . . . .  | 111 |
| <a href="#">Langmuir::XYZWriter</a>  |     |
| A class to output xyz files . . . . .  | 126 |



## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

|                    |     |
|--------------------|-----|
| agent.h            | 129 |
| chargeagent.cpp    | 129 |
| chargeagent.h      | 130 |
| checkpointer.cpp   | 130 |
| checkpointer.h     | 131 |
| copy_to_frank.py   | 131 |
| cubicgrid.cpp      | 131 |
| cubicgrid.h        | 132 |
| drainagent.cpp     | 132 |
| drainagent.h       | 133 |
| fluxagent.cpp      | 133 |
| fluxagent.h        | 133 |
| gridview.cpp       | 134 |
| keyvalueparser.cpp | 134 |
| keyvalueparser.h   | 134 |
| langmuir.cpp       | 135 |
| langmuirview.cpp   | 135 |
| openclhelper.cpp   | 136 |
| openclhelper.h     | 136 |
| output.cpp         | 136 |
| output.h           | 137 |
| parameters.h       | 138 |
| pbsgpuparser.cpp   | 138 |
| pbsgpuparser.h     | 139 |
| potential.cpp      | 139 |
| potential.h        | 139 |
| rand.cpp           | 140 |
| rand.h             | 140 |
| simulation.cpp     | 141 |
| simulation.h       | 141 |
| sourceagent.cpp    | 141 |
| sourceagent.h      | 142 |
| test.cpp           | 142 |
| tolerance.cpp      | 142 |
| tolerance.h        | 143 |
| variable.h         | 143 |
| world.cpp          | 144 |
| world.h            | 144 |

|                            |     |
|----------------------------|-----|
| <a href="#">writer.cpp</a> | 145 |
| <a href="#">writer.h</a>   | 145 |

## Chapter 6

# Namespace Documentation

### 6.1 `copy_to_frank` Namespace Reference

#### Variables

- tuple `work` = `os.getcwd()`
- list `exclude`
- list `found` = []
- `copy_me` = `True`

#### 6.1.1 Variable Documentation

6.1.1.1 `copy_to_frank.copy_me` = `True`

6.1.1.2 list `copy_to_frank.exclude`

#### Initial value:

```
1 [
2     re.compile(r'CMake.*'),
3     re.compile(r'.*\.py')
4 ]
```

6.1.1.3 list `copy_to_frank.found` = []

6.1.1.4 tuple `copy_to_frank.work` = `os.getcwd()`

### 6.2 Langmuir Namespace Reference

#### Classes

- class `Agent`  
*A class that abstractly represents an object that can occupy grid sites.*
- class `ChargeAgent`  
*A class to represent moving charged particles.*
- class `ElectronAgent`  
*A class to represent moving negative charges.*
- class `HoleAgent`

- A class to represent moving positive charges.*

  - class [CheckPointer](#)

*A class to read and write checkpoint files.*
  - class [Grid](#)

*A class to hold Agents, calculate their positions, and store the background potential.*
  - class [DrainAgent](#)

*A class to remove charges.*
  - class [ElectronDrainAgent](#)

*A class to remove ElectronAgents.*
  - class [HoleDrainAgent](#)

*A class to remove HoleAgents.*
  - class [RecombinationAgent](#)

*A class to remove Excitons.*
  - class [FluxAgent](#)

*A class to change the number of carriers in the system.*
  - class [KeyValueParser](#)

*A class to read the parameters and store them in the correct place.*
  - 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).*
  - struct [ConfigurationInfo](#)

*A struct to temporarily store site IDs.*
  - struct [SimulationParameters](#)

*A struct to store all simulation options To add new variables, follow these steps:*
  - class [PBSGPUParser](#)
  - class [Potential](#)

*A class to calculate the potential.*
  - class [Random](#)

*A class to generate random numbers.*
  - class [Simulation](#)

*A class to orchestrate the calculation.*
  - class [SourceAgent](#)

*A class to inject charges.*
  - class [ElectronSourceAgent](#)

*A class to inject ElectronAgents.*
  - class [HoleSourceAgent](#)

*A class to inject HoleAgents.*
  - class [ExcitonSourceAgent](#)

*A class to inject Excitons.*
  - class [Tolerance](#)

*A class to check if the simulation is converging.*
  - class [Variable](#)

*A class to map between variable names (keys) and locations (references)*
  - class [TypedVariable](#)

*A template 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.*
- class [FluxWriter](#)
  - A class to output source and drain info.*
- class [CarrierWriter](#)
  - A class to output carrier stats (lifetime and pathlength)*
- class [ExcitonWriter](#)
  - A class to output exciton stats (lifetime and pathlength)*
- class [GridImage](#)
  - A class to draw images of the grid.*
- class [Logger](#)
  - A class that organizes output.*

## 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.*
- std::ostream & [operator<<](#) (std::ostream &stream, const [KeyValueParser](#) &keyValueParser)
- 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*
- QDataStream & [operator<<](#) (QDataStream &stream, [Random](#) &random)
- QDataStream & [operator>>](#) (QDataStream &stream, [Random](#) &random)
- QTextStream & [operator<<](#) (QTextStream &stream, [Random](#) &random)
- QTextStream & [operator>>](#) (QTextStream &stream, [Random](#) &random)
- std::ostream & [operator<<](#) (std::ostream &stream, [Random](#) &random)
- std::istream & [operator>>](#) (std::istream &stream, [Random](#) &random)
- 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.*

## 6.2.1 Function Documentation

### 6.2.1.1 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 examining existing files in path with a similar form (path/file.current\_date.a\_number). The file is renamed, not copied.

#### Parameters

|             |                                       |
|-------------|---------------------------------------|
| <i>name</i> | 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

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

check the parameters, making sure they are valid

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

output QDateTime as qint64 mSecsSinceEpoch

### 6.2.1.4 QDataStream & Langmuir::operator<< ( QDataStream & *stream*, Random & *random* )

#### Warning

This may not quite be working correctly

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

Output [Agent](#) type enum to stream.

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

Output [Agent](#) type enum to debug information.

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

### 6.2.1.8 std::ostream & Langmuir::operator<< ( std::ostream & *stream*, const KeyValuePair & *keyValuePair* )

### 6.2.1.9 QTextStream & Langmuir::operator<< ( QTextStream & *stream*, Random & *random* )

#### Warning

This may not quite be working correctly

6.2.1.10 `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.

6.2.1.11 `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.

6.2.1.12 `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.

6.2.1.13 `std::ostream& Langmuir::operator<< ( std::ostream & stream, Random & random )`

6.2.1.14 `QTextStream & Langmuir::operator<< ( QTextStream & stream, const Grid::CubeFace e )`

Overload QTextStream for the [Grid::CubeFace](#) Enum.

6.2.1.15 `QDebug Langmuir::operator<< ( QDebug dbg, const Grid::CubeFace e )`

Overload QDebug for the [Grid::CubeFace](#) Enum.

6.2.1.16 `QDataStream& Langmuir::operator>> ( QDataStream & stream, Random & random )`

Warning

This may not quite be working correctly

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

6.2.1.18 `QTextStream& Langmuir::operator>> ( QTextStream & stream, Random & random )`

Warning

This may not quite be working correctly

6.2.1.19 `std::istream& Langmuir::operator>> ( std::istream & stream, Random & random )`

6.2.1.20 `void Langmuir::setCalculatedValues ( SimulationParameters & par )` `[inline]`

sets parameters that depend upon other parameters





## Chapter 7

# Class Documentation

### 7.1 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.

### 7.1.1 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.

### 7.1.2 Member Enumeration Documentation

#### 7.1.2.1 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.

### 7.1.3 Constructor & Destructor Documentation

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

Create an `Agent`.

## Parameters

|               |  |
|---------------|--|
| <i>type</i>   | identifier enum<br>• for example: Electron, Hole, etc. |
| <i>world</i>  | reference world object                                 |
| <i>site</i>   | grid site id <a href="#">Agent</a> occupies            |
| <i>parent</i> | parent QObject   |

7.1.3.2 `Langmuir::Agent::~~Agent ( ) [inline],[virtual]`

Destroy [Agent](#).

## 7.1.4 Member Function Documentation

7.1.4.1 `int Langmuir::Agent::getCurrentSite ( ) const [inline]`

Get [Agent](#) current site.

7.1.4.2 `int Langmuir::Agent::getFutureSite ( ) const [inline]`

Get [Agent](#) future site.

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

Get [Agent](#) neighbor list.

7.1.4.4 `Agent::Type Langmuir::Agent::getType ( ) const [inline]`

Get [Agent::Type](#) enum.

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

Get [Langmuir::World](#) reference.

7.1.4.6 `void Langmuir::Agent::setCurrentSite ( int site ) [inline]`

Set [Agent](#) current site.

7.1.4.7 `void Langmuir::Agent::setFutureSite ( int site ) [inline]`

Set [Agent](#) future site.

7.1.4.8 `void Langmuir::Agent::setNeighbors ( QVector< int > neighbors ) [inline]`

Set [Agent](#) neighbor list.

7.1.4.9 `QString Langmuir::Agent::toQString ( const Agent::Type e ) [inline],[static]`

Convert [Agent](#) type enum to QString.

### 7.1.5 Member Data Documentation

#### 7.1.5.1 `int Langmuir::Agent::m_fSite` [protected]

Future site the [Agent](#) will occupy.

#### 7.1.5.2 `QVector<int> Langmuir::Agent::m_neighbors` [protected]

List fo neighboring site ids.

#### 7.1.5.3 `int Langmuir::Agent::m_site` [protected]

Current site the [Agent](#) occupies.

#### 7.1.5.4 `Type Langmuir::Agent::m_type` [protected]

[Agent](#) Type enum.

#### 7.1.5.5 `World& Langmuir::Agent::m_world` [protected]

Reference to [World](#) object.

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

- [agent.h](#)

## 7.2 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*

#### 7.2.1 Detailed Description

A class to output carrier stats (lifetime and pathlength)

## 7.2.2 Constructor & Destructor Documentation

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

constructs the writer, has the same parameters as [OutputInfo](#)

## 7.2.3 Member Function Documentation

### 7.2.3.1 void Langmuir::CarrierWriter::write ( ChargeAgent & charge )

write the charge carrier statistics to the stream

## 7.2.4 Member Data Documentation

### 7.2.4.1 OutputStream Langmuir::CarrierWriter::m\_stream [protected]

output file stream

### 7.2.4.2 World& Langmuir::CarrierWriter::m\_world [protected]

reference to the world object

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

- [writer.h](#)
- [writer.cpp](#)

## 7.3 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 existence.*
- 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

### 7.3.1 Detailed Description

A class to represent moving charged particles.

### 7.3.2 Constructor & Destructor Documentation

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

Construct charge.

[ChargeAgent](#)

#### Parameters

|                |  |
|----------------|--|
| <i>getType</i> | Agent type; must be <a href="#">Agent::Electron</a> or <a href="#">Agent::Hole</a> |
| <i>world</i>   | reference to world   |
| <i>grid</i>    | reference to grid  |
| <i>site</i>    | site id in grid  |
| <i>parent</i>  | parent QObject   |

**7.3.2.2** `Langmuir::ChargeAgent::~~ChargeAgent ( )` [virtual]

Destroy charge.

### 7.3.3 Member Function Documentation

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

Calculate the exciton binding energy.

#### Parameters

|             |   |
|-------------|---|
| <i>site</i> | the site to check in other <a href="#">Grid</a> |
|-------------|---|

#### Returns

- +0.5 eV if exciton
- 0 otherwise

Implemented in [Langmuir::HoleAgent](#), and [Langmuir::ElectronAgent](#).

**7.3.3.2** `int Langmuir::ChargeAgent::charge ( )`

Get the charge of the [ChargeAgent](#).

**7.3.3.3** `void Langmuir::ChargeAgent::chooseFuture ( )`

Propose a random site to move to.

**7.3.3.4** `void Langmuir::ChargeAgent::compareCoulomb ( )`

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

**7.3.3.5** `void Langmuir::ChargeAgent::completeTick ( )`

Perform action, called after decideFuture.

### 7.3.3.6 void Langmuir::ChargeAgent::coulombCPU ( )

Calculate the Coulomb potential on the CPU.

#### Note

The result is stored in `m_de`

### 7.3.3.7 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

### 7.3.3.8 double Langmuir::ChargeAgent::coulombInteraction ( )

Perform [coulombCPU\(\)](#) or [coulombGPU\(\)](#)

depends upon [SimulationParameters::useOpenCL](#) and [SimulationParameters::okCL](#)

#### Returns

[ChargeAgent::m\\_de](#)

### 7.3.3.9 void Langmuir::ChargeAgent::decideFuture ( )

Decide what should happen, called after `chooseFuture`.

### 7.3.3.10 Grid & Langmuir::ChargeAgent::getGrid ( )

Get the grid this [ChargeAgent](#) exists in.

### 7.3.3.11 int Langmuir::ChargeAgent::getOpenCLID ( )

Get the [ChargeAgent](#) OpenCL identifier.

#### See Also

[OpenCIHelper](#)

### 7.3.3.12 int Langmuir::ChargeAgent::lifetime ( )

Number of steps [ChargeAgent](#) has existed.



7.3.3.13 `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](#).

7.3.3.14 `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](#).

7.3.3.15 `int Langmuir::ChargeAgent::pathlength ( )`

Number of sites [ChargeAgent](#) has traversed.

7.3.3.16 `bool Langmuir::ChargeAgent::removed ( )`

True if decideFuture removed the charge from the grid.

7.3.3.17 `void Langmuir::ChargeAgent::setOpenCLID ( int id )`

Set the [ChargeAgent](#) OpenCL identifier.

See Also

[OpenCIHelper](#)

7.3.3.18 `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

## 7.3.4 Member Data Documentation

7.3.4.1 `int Langmuir::ChargeAgent::m_charge [protected]`

Charge of [ChargeAgent](#) (in units of e)

7.3.4.2 `double Langmuir::ChargeAgent::m_de [protected]`

The difference in Coulomb potential between [ChargeAgent::m\\_site](#) and [ChargeAgent::m\\_fSite](#).

#### 7.3.4.3 `Grid& Langmuir::ChargeAgent::m_grid` [protected]

The [Grid](#) the [ChargeAgent](#) lives in.

#### 7.3.4.4 `int Langmuir::ChargeAgent::m_lifetime` [protected]

Number of steps [ChargeAgent](#) as been in existence.

#### 7.3.4.5 `int Langmuir::ChargeAgent::m_openClID` [protected]

The index of the Charge in the OpenCL vectors (see [OpenClHelper](#))

#### 7.3.4.6 `int Langmuir::ChargeAgent::m_pathlength` [protected]

Number of grid spaces [ChargeAgent](#) has moved.

#### 7.3.4.7 `bool Langmuir::ChargeAgent::m_removed` [protected]

Removed status of [ChargeAgent](#).

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

- [chargeagent.h](#)
- [chargeagent.cpp](#)

## 7.4 [Langmuir::Checkpoint](#) Class Reference

A class to read and write checkpoint files.

```
#include <checkpoint.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

- [Checkpoint](#) ([World](#) &world, `QObject *parent=0`)  
*Create the checkpoint 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*

### 7.4.1 Detailed Description

A class to read and write checkpoint files.

Checkpoint files are essentially the same as input files

## 7.4.2 Member Enumeration Documentation

### 7.4.2.1 enum Langmuir::CheckPointer::Section

A way to identify different sections in the input file.

Enumerator:

***Parameters***

***Electrons***

***Holes***

***Defects***

***Traps***

***TrapPotentials***

***RandomState***

***FluxState***

## 7.4.3 Constructor & Destructor Documentation

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

Create the checkpointer object.

Parameters

|               |                        |
|---------------|------------------------|
| <i>world</i>  | reference world object |
| <i>parent</i> | parent QObject         |

## 7.4.4 Member Function Documentation

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

check to see if input stream has failed

Parameters

|                |  |
|----------------|--|
| <i>stream</i>  | input stream                                 |
| <i>message</i> | the error message to output if stream failed |

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

load simulation information

Parameters

|                   |   |
|-------------------|---|
| <i>fileName</i>   | name of input file                          |
| <i>configInfo</i> | temporary storage for electrons, holes, etc |

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

load defect sites from input file

## Parameters

|                   |                                |
|-------------------|--------------------------------|
| <i>stream</i>     | the input stream               |
| <i>configInfo</i> | temporary storage for site ids |

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

load electrons sites from input file

## Parameters

|                   |                                |
|-------------------|--------------------------------|
| <i>stream</i>     | the input stream               |
| <i>configInfo</i> | temporary storage for site ids |

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

load flux state from input file

## Parameters

|                   |                                  |
|-------------------|----------------------------------|
| <i>stream</i>     | the input stream                 |
| <i>configInfo</i> | temporary storage for flux state |

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

load hole sites from input file

## Parameters

|                   |                                |
|-------------------|--------------------------------|
| <i>stream</i>     | the input stream               |
| <i>configInfo</i> | temporary storage for site ids |

7.4.4.7 `std::istream & Langmuir::CheckPointer::loadParameters ( std::istream & stream )` [private]

load parameter from input file

## Parameters

|               |                  |
|---------------|------------------|
| <i>stream</i> | the input stream |
|---------------|------------------|

7.4.4.8 `std::istream & Langmuir::CheckPointer::loadRandomState ( std::istream & stream )` [private]

load random number generator state from input file

## Parameters

|               |                  |
|---------------|------------------|
| <i>stream</i> | the input stream |
|---------------|------------------|

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

load trap energies from input file

#### Parameters

|                   |                                     |
|-------------------|-------------------------------------|
| <i>stream</i>     | the input stream                    |
| <i>configInfo</i> | temporary storage for site energies |

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

load trap sites from input file

#### Parameters

|                   |                                |
|-------------------|--------------------------------|
| <i>stream</i>     | the input stream               |
| <i>configInfo</i> | temporary storage for site ids |

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

save simulation information

#### Parameters

|                 |                     |
|-----------------|---------------------|
| <i>fileName</i> | name of output file |
|-----------------|---------------------|

7.4.4.12 `std::ostream & Langmuir::CheckPointer::saveDefects ( std::ostream & stream )` [private]

save defect site ids to output file

#### Parameters

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

7.4.4.13 `std::ostream & Langmuir::CheckPointer::saveElectrons ( std::ostream & stream )` [private]

save electron site ids to output file

#### Parameters

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

7.4.4.14 `std::ostream & Langmuir::CheckPointer::saveFluxState ( std::ostream & stream )` [private]

save flux states to output file

#### Parameters

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

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

save hole site ids to output file

**Parameters**

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

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

save parameters to output file

**Parameters**

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

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

save random number generator state to output file

**Parameters**

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

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

save trap energies to output file

**Parameters**

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

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

save trap site ids to output file

**Parameters**

|               |               |
|---------------|---------------|
| <i>stream</i> | output stream |
|---------------|---------------|

**7.4.5 Member Data Documentation****7.4.5.1** `World& Langmuir::CheckPointer::m_world` `[private]`

reference to world object

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

- [checkpointer.h](#)
- [checkpointer.cpp](#)

## 7.5 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*

### 7.5.1 Detailed Description

A struct to temporarily store site IDs.

### 7.5.2 Member Data Documentation

#### 7.5.2.1 `QList<qint32> Langmuir::ConfigurationInfo::defects`

a list of current defects site IDs

#### 7.5.2.2 `QList<qint32> Langmuir::ConfigurationInfo::electrons`

a list of current electron site IDs

#### 7.5.2.3 `QList<quint64> Langmuir::ConfigurationInfo::fluxInfo`

a list of flux attempt, success values

#### 7.5.2.4 `QList<qint32> Langmuir::ConfigurationInfo::holes`

a list of current holes site IDs

#### 7.5.2.5 `QList<qreal> Langmuir::ConfigurationInfo::trapPotentials`

a list of current traps site IDs



## 7.5.2.6 QList&lt;qint32&gt; Langmuir::ConfigurationInfo::traps

a list of current traps site IDs

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

- [parameters.h](#)

## 7.6 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

#### 7.6.1 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.

#### 7.6.2 Constructor & Destructor Documentation

7.6.2.1 Langmuir::DrainAgent::DrainAgent ( [World](#) & world, [Grid](#) & grid, [QObject](#) \* parent = 0 )

create a [DrainAgent](#)

#### 7.6.3 Member Function Documentation

7.6.3.1 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 files:

- [drainagent.h](#)
- [drainagent.cpp](#)

## 7.7 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

### 7.7.1 Detailed Description

A class to represent moving negative charges.

### 7.7.2 Constructor & Destructor Documentation

7.7.2.1 [Langmuir::ElectronAgent::ElectronAgent](#) ( [World](#) & *world*, int *site*, QObject \* *parent* = 0 )

Construct [ElectronAgent](#).

### 7.7.3 Member Function Documentation

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

Calculate Exciton Binding Energy.

#### Parameters

|             |   |
|-------------|---|
| <i>site</i> | the site to check in other <a href="#">Grid</a> |
|-------------|---|

#### Returns

- +0.5 eV if exciton
- 0 otherwise

Implements [Langmuir::ChargeAgent](#).

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

Return other [Grid](#).

#### Returns

[World::holeGrid](#)

Implements [Langmuir::ChargeAgent](#).

### 7.7.3.3 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 files:

- [chargeagent.h](#)
- [chargeagent.cpp](#)

## 7.8 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

#### 7.8.1 Detailed Description

A class to remove ElectronAgents.

#### 7.8.2 Constructor & Destructor Documentation

##### 7.8.2.1 Langmuir::ElectronDrainAgent::ElectronDrainAgent ( [World](#) & world, int site, QObject \* parent = 0 )

create an [ElectronDrainAgent](#) at a specific site

##### 7.8.2.2 Langmuir::ElectronDrainAgent::ElectronDrainAgent ( [World](#) & world, [Grid::CubeFace](#) cubeFace, QObject \* parent = 0 )

create a [ElectronDrainAgent](#) at a specific [Grid::CubeFace](#)

### 7.8.3 Member Function Documentation

#### 7.8.3.1 `double Langmuir::ElectronDrainAgent::energyChange ( int fSite )` `[protected]`, `[virtual]`

same as [FluxAgent::energyChange\(\)](#), but specialized for ElectronAgents.

Note really used because the default [FluxAgent::shouldTransport\(\)](#) behavior, 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 files:

- [drainagent.h](#)
- [drainagent.cpp](#)

## 7.9 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

#### 7.9.1 Detailed Description

A class to inject ElectronAgents.

#### 7.9.2 Constructor & Destructor Documentation

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

create an [ElectronSourceAgent](#) at a specific site

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

create an [ElectronSourceAgent](#) at a specific [Grid::CubeFace](#)

### 7.9.3 Member Function Documentation

7.9.3.1 `double Langmuir::ElectronSourceAgent::energyChange ( int site )` [protected],[virtual]

same as [FluxAgent::energyChange\(\)](#), but specialized for ElectronAgents.

Reimplemented from [Langmuir::FluxAgent](#).

7.9.3.2 `void Langmuir::ElectronSourceAgent::inject ( int site )` [protected],[virtual]

same as [SourceAgent::inject\(\)](#), but specialized for ElectronAgents.

Implements [Langmuir::SourceAgent](#).

7.9.3.3 `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 files:

- [sourceagent.h](#)
- [sourceagent.cpp](#)

## 7.10 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

#### 7.10.1 Detailed Description

A class to inject Excitons.

## 7.10.2 Constructor & Destructor Documentation

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

create an [ExcitonSourceAgent](#)

## 7.10.3 Member Function Documentation

7.10.3.1 `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](#).

7.10.3.2 `double Langmuir::ExcitonSourceAgent::energyChange ( int site )` `[protected]`, `[virtual]`

currently implemented as zero and not really used

Reimplemented from [Langmuir::FluxAgent](#).

7.10.3.3 `void Langmuir::ExcitonSourceAgent::inject ( int site )` `[protected]`, `[virtual]`

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

Implements [Langmuir::SourceAgent](#).

7.10.3.4 `bool Langmuir::ExcitonSourceAgent::shouldTransport ( int site )` `[protected]`, `[virtual]`

uses the simple constant probability method

Reimplemented from [Langmuir::SourceAgent](#).

7.10.3.5 `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 files:

- [sourceagent.h](#)
- [sourceagent.cpp](#)

## 7.11 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*

### 7.11.1 Detailed Description

A class to output exciton stats (lifetime and pathlength)

### 7.11.2 Constructor & Destructor Documentation

7.11.2.1 [Langmuir::ExcitonWriter::ExcitonWriter](#) ( [World](#) & *world*, const [QString](#) & *name*, [QObject](#) \* *parent* = 0 )

constructs the writer, has the same parameters as [OutputInfo](#)

### 7.11.3 Member Function Documentation

7.11.3.1 void [Langmuir::ExcitonWriter::write](#) ( [ChargeAgent](#) & *charge1*, [ChargeAgent](#) & *charge2*, bool *recombined* = false )

write the exciton statistics to the stream

### 7.11.4 Member Data Documentation

7.11.4.1 [OutputStream](#) [Langmuir::ExcitonWriter::m\\_stream](#) [protected]

output file stream

7.11.4.2 [World&](#) [Langmuir::ExcitonWriter::m\\_world](#) [protected]

reference to the world object

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

- [writer.h](#)
- [writer.cpp](#)

## 7.12 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 probability of success*
- double [successRate](#) () const  
*calculate and return the current rate of success*
- double [successProbabilitySinceLast](#) () const  
*calculate and return the probability 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 behavior 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

### 7.12.1 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`)

### 7.12.2 Constructor & Destructor Documentation

#### 7.12.2.1 Langmuir::FluxAgent::FluxAgent ( `Agent::Type` type, `World` & world, `Grid` & grid, `QObject` \* parent = 0 )

Create the flux agent.

#### Parameters

|               |  |
|---------------|--|
| <i>type</i>   | either a <code>Agent::Source</code> or <code>Agent::Drain</code> |
| <i>world</i>  | reference to world object  |
| <i>grid</i>   | reference to grid  |
| <i>parent</i> | parent <code>QObject</code>                                      |

### 7.12.2.2 `Langmuir::FluxAgent::~~FluxAgent ( )`

unregisters [FluxAgent](#) from the grid

## 7.12.3 Member Function Documentation

### 7.12.3.1 `unsigned long int Langmuir::FluxAgent::attempts ( ) const`

get the [FluxAgent](#)'s attempt counter

### 7.12.3.2 `unsigned long int Langmuir::FluxAgent::attemptsSinceLast ( ) const`

get the number of attempts since `storeLast()` was called

### 7.12.3.3 `double Langmuir::FluxAgent::energyChange ( int site )` `[protected]`, `[virtual]`

The energy change associated with moving a carrier from the [FluxAgent](#) to a site.

#### Parameters

|             |                   |
|-------------|-------------------|
| <i>site</i> | the site involved |
|-------------|-------------------|

#### Returns

the energy change

Reimplemented in [Langmuir::ExcitonSourceAgent](#), [Langmuir::HoleSourceAgent](#), [Langmuir::ElectronSourceAgent](#), [Langmuir::HoleDrainAgent](#), and [Langmuir::ElectronDrainAgent](#).

### 7.12.3.4 `Grid::CubeFace Langmuir::FluxAgent::face ( ) const`

get the [Grid::CubeFace](#) this [FluxAgent](#) is assigned to

### 7.12.3.5 `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.

### 7.12.3.6 `Grid & Langmuir::FluxAgent::grid ( ) const`

get the [Grid](#) this [FluxAgent](#) belongs to

### 7.12.3.7 `void Langmuir::FluxAgent::initializeSite ( int site )` `[protected]`

assign the [FluxAgent](#) to a specific site in the grid

#### Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site in the grid |
|-------------|----------------------|

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

assign the [FluxAgent](#) to a specific [Grid::CubeFace](#)

#### Parameters

|                 |  |
|-----------------|--|
| <i>cubeFace</i> | the face of a cubic grid; for example <a href="#">Grid::Left</a> |
|-----------------|--|

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.

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

get the [FluxAgent](#)'s potential

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

get the [FluxAgent](#)'s rate

7.12.3.11 void Langmuir::FluxAgent::resetCounters ( )

set the attempt and success counters to zero

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

set the [FluxAgent](#)'s attempt counter

#### Parameters

|                  |                                  |
|------------------|----------------------------------|
| <i>potential</i> | the value of the attempt counter |
|------------------|----------------------------------|

#### Warning

also calls [storeLast\(\)](#)

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

set the [FluxAgent](#)'s potential

#### Parameters

|                  |                            |
|------------------|----------------------------|
| <i>potential</i> | the value of the potential |
|------------------|----------------------------|

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

set the [FluxAgent](#)'s rate

#### Parameters

|                  |                       |
|------------------|-----------------------|
| <i>potential</i> | the value of the rate |
|------------------|-----------------------|

7.12.3.15 `void Langmuir::FluxAgent::setRateSmartly ( double rate, double dflt )`

set the [FluxAgent](#)'s rate

#### Parameters

|                  |                                      |
|------------------|--------------------------------------|
| <i>potential</i> | the value of the rate                |
| <i>dflt</i>      | the default value to set the rate to |

If rate is negative, uses the default rate instead

7.12.3.16 `void Langmuir::FluxAgent::setSuccesses ( unsigned long int value )`

set the [FluxAgent](#)'s success counter

#### Parameters

|                  |                          |
|------------------|--------------------------|
| <i>potential</i> | the value of the counter |
|------------------|--------------------------|

#### Warning

also calls [storeLast\(\)](#)

7.12.3.17 `bool Langmuir::FluxAgent::shouldTransport ( int site )` `[protected], [virtual]`

decide if the [FluxAgent](#) should transport a carrier to/from a given site

#### Parameters

|             |                   |
|-------------|-------------------|
| <i>site</i> | the site involved |
|-------------|-------------------|

#### Returns

true if the [FluxAgent](#) should transport to/from the site

The default behaviour is for the [FluxAgent](#) to use a simple constant probability 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](#).

7.12.3.18 `unsigned long int Langmuir::FluxAgent::stepsSinceLast ( ) const`

get the number of steps since [storeLast\(\)](#) was called

7.12.3.19 `void Langmuir::FluxAgent::storeLast ( )`

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

7.12.3.20 `unsigned long int Langmuir::FluxAgent::successes ( ) const`

get the [FluxAgent](#)'s success counter

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

get the number of successes since [storeLast\(\)](#) was called

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

calculate and return the current probability 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.

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

calculate and return the probability of success since [storeLast\(\)](#) was called

This is the number of [successesSinceLast\(\)](#) divided by the number of [attemptsSinceLast\(\)](#) (x100).

7.12.3.24 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.

7.12.3.25 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.

## 7.12.4 Member Data Documentation

7.12.4.1 unsigned long int Langmuir::FluxAgent::m\_attempts [protected]

the number of times the [FluxAgent](#) has tried to transport.

7.12.4.2 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.

7.12.4.3 Grid& Langmuir::FluxAgent::m\_grid [protected]

the grid this [FluxAgent](#) resides in

7.12.4.4 unsigned long int Langmuir::FluxAgent::m\_lastAttempts [protected]

storage to note the number of successes at some step

7.12.4.5 unsigned long int Langmuir::FluxAgent::m\_lastStep [protected]

the step at which last was noted

#### 7.12.4.6 unsigned long int Langmuir::FluxAgent::m\_lastSuccesses [protected]

storage to note the number of successes at some step

#### 7.12.4.7 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.

#### 7.12.4.8 double Langmuir::FluxAgent::m\_probability [protected]

the constant probability used in the default behaviour of [shouldTransport\(\)](#).

#### 7.12.4.9 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 files:

- [fluxagent.h](#)
- [fluxagent.cpp](#)

## 7.13 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*

#### 7.13.1 Detailed Description

A class to output source and drain info.

#### 7.13.2 Constructor & Destructor Documentation

##### 7.13.2.1 Langmuir::FluxWriter::FluxWriter ( [World](#) & *world*, const QString & *name*, QObject \* *parent* = 0 )

constructs the writer, has the same parameters as [OutputInfo](#)

### 7.13.3 Member Function Documentation

#### 7.13.3.1 void Langmuir::FluxWriter::write ( )

write the flux statistics of the current step to the stream

### 7.13.4 Member Data Documentation

#### 7.13.4.1 OutputStream Langmuir::FluxWriter::m\_stream [protected]

output file stream

#### 7.13.4.2 World& Langmuir::FluxWriter::m\_world [protected]

reference to the world object

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

- [writer.h](#)
- [writer.cpp](#)

## 7.14 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.*

### 7.14.1 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**.

### 7.14.2 Member Enumeration Documentation

#### 7.14.2.1 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

### 7.14.3 Constructor & Destructor Documentation

#### 7.14.3.1 `Langmuir::Grid::Grid ( World & world, QObject * parent = 0 )`

Create a grid.

Parameters

|               |                               |
|---------------|-------------------------------|
| <i>world</i>  | reference to the world object |
| <i>parent</i> | QObject this belongs to       |

### 7.14.3.2 Langmuir::Grid::~~Grid ( )

Destroy the grid.

## 7.14.4 Member Function Documentation

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

Add some value to the background potential at a site.

#### Parameters

|                  |                  |
|------------------|------------------|
| <i>site</i>      | the "s-site ID"  |
| <i>potential</i> | the value to add |

### 7.14.4.2 Agent \* Langmuir::Grid::agentAddress ( int *site* )

Get a pointer to the [Agent](#) at a site.

#### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

#### Warning

may be NULL if there is no [Agent](#)

### 7.14.4.3 Agent::Type Langmuir::Grid::agentType ( int *site* )

Get the type of [Agent](#) at a site.

#### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

#### Warning

if there is no [Agent](#), it should be [Agent::Empty](#)

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

Get the serial site ID.

#### Parameters

|               |           |
|---------------|-----------|
| <i>xIndex</i> | x site ID |
| <i>yIndex</i> | y site ID |
| <i>zIndex</i> | 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".

#### 7.14.4.5 `int Langmuir::Grid::getIndexX ( int site )`

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

##### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

The y-site ID can be thought of as the x-value of the corner of a [Grid](#) site.

##### See Also

[getIndexS](#)

#### 7.14.4.6 `int Langmuir::Grid::getIndexY ( int site )`

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

##### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

The y-site ID can be thought of as the y-value of the corner of a [Grid](#) site.

##### See Also

[getIndexS](#)

#### 7.14.4.7 `int Langmuir::Grid::getIndexZ ( int site )`

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

##### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

The y-site ID can be thought of as the z-value of the corner of a [Grid](#) site.

##### See Also

[getIndexS](#)

#### 7.14.4.8 `double Langmuir::Grid::getPositionX ( int site )`

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

##### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | 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.

#### 7.14.4.9 double Langmuir::Grid::getPositionY ( int *site* )

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

##### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | 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.

#### 7.14.4.10 double Langmuir::Grid::getPositionZ ( int *site* )

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

##### Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | 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.

#### 7.14.4.11 QList< Agent \* > & Langmuir::Grid::getSpecialAgentList ( Grid::CubeFace *cubeFace* )

Get a list of special Agents assigned to a specific [Grid::CubeFace](#).

##### Parameters

|                 |                                      |
|-----------------|--------------------------------------|
| <i>cubeFace</i> | the face of the <a href="#">Grid</a> |
|-----------------|--------------------------------------|

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

Calculate the neighboring sites of a given face of the [Grid](#).

##### Parameters

|                 |  |
|-----------------|--|
| <i>cubeFace</i> | the face of the <a href="#">Grid</a> to consider |
|-----------------|--|

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

Calculate the neighboring sites of a given site.

##### Parameters

|                     |   |
|---------------------|---|
| <i>site</i>         | the "s-site ID"   |
| <i>hoppingRange</i> | the number of adjacent sites to consider in the calculation |

#### 7.14.4.14 double Langmuir::Grid::potential ( int *site* )

Get the background potential at some site.

## Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

## 7.14.4.15 void Langmuir::Grid::registerAgent ( Agent \* agent )

Assign an [Agent](#) to a site in the [Grid](#).

## Parameters

|              |  |
|--------------|--|
| <i>agent</i> | a pointer to the <a href="#">Agent</a> |
|--------------|--|

## 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](#).

## 7.14.4.16 void Langmuir::Grid::registerDefect ( int site )

Assign a site to be [Agent::Defect](#).

## Parameters

|             |  |
|-------------|--|
| <i>site</i> |  |
|-------------|--|

## 7.14.4.17 void Langmuir::Grid::registerSpecialAgent ( Agent \* agent, Grid::CubeFace cubeFace )

Assign an [Agent](#) to a special location.

## Parameters

|                 |  |
|-----------------|--|
| <i>agent</i>    | a pointer to the <a href="#">Agent</a> |
| <i>cubeFace</i> | the face of the <a href="#">Grid</a>   |

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

## 7.14.4.18 void Langmuir::Grid::setPotential ( int site, double potential )

Set the background potential at a site to some value.

## Parameters

|                  |                  |
|------------------|------------------|
| <i>site</i>      | the "s-site ID"  |
| <i>potential</i> | the value to set |

## 7.14.4.19 QVector&lt; int &gt; 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

|           |                    |
|-----------|--------------------|
| <i>xi</i> | starting x-site ID |
| <i>xf</i> | stopping x-site ID |
| <i>yi</i> | starting y-site ID |
| <i>yf</i> | stopping y-site ID |
| <i>zi</i> | starting z-site ID |
| <i>zf</i> | stopping z-site ID |

7.14.4.20 `int Langmuir::Grid::specialAgentCount ( )`

The total number of special Agents.

7.14.4.21 `QString Langmuir::Grid::toString ( const Grid::CubeFace e ) [static]`

7.14.4.22 `double Langmuir::Grid::totalDistance ( int site1, int site2 )`

Get the distance between two sites.

## Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

7.14.4.23 `void Langmuir::Grid::unregisterAgent ( Agent * agent )`

Remove an [Agent](#) from the [Grid](#).

## Parameters

|              |  |
|--------------|--|
| <i>agent</i> | a pointer to the <a href="#">Agent</a> |
|--------------|--|

7.14.4.24 `void Langmuir::Grid::unregisterDefect ( int site )`

Remove a defect from the [Grid](#).

## Parameters

|             |                 |
|-------------|-----------------|
| <i>site</i> | the "s-site ID" |
|-------------|-----------------|

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

Remove an [Agent](#) from the special list of Agents in the [Grid](#).

## Parameters

|                 |  |
|-----------------|--|
| <i>agent</i>    | a pointer to the <a href="#">Agent</a> |
| <i>cubeFace</i> | the face of the <a href="#">Grid</a>   |

7.14.4.26 `int Langmuir::Grid::volume ( )`

Get the total number of sites.

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

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

##### Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

#### 7.14.4.28 int Langmuir::Grid::xDistancei ( int *site1*, int *site2* )

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

##### Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

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

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

##### Parameters

|              |                             |
|--------------|-----------------------------|
| <i>site1</i> | the first site              |
| <i>site2</i> | 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).

#### 7.14.4.30 int Langmuir::Grid::xImageDistancei ( int *site1*, int *site2* )

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

##### Parameters

|              |                             |
|--------------|-----------------------------|
| <i>site1</i> | the first site              |
| <i>site2</i> | 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).

#### 7.14.4.31 int Langmuir::Grid::xSize ( )

Get the number of sites along the x-direction.

#### 7.14.4.32 int Langmuir::Grid::xyPlaneArea ( )

Get the number of sites in the xy-plane.

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

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



## Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

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

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

## Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

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

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

## Parameters

|              |                             |
|--------------|-----------------------------|
| <i>site1</i> | the first site              |
| <i>site2</i> | 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).

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

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

## Parameters

|              |                             |
|--------------|-----------------------------|
| <i>site1</i> | the first site              |
| <i>site2</i> | 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).

## 7.14.4.37 int Langmuir::Grid::ySize ( )

Get the number of sites along the y-direction.

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

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

## Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

#### 7.14.4.39 int Langmuir::Grid::zDistancei ( int *site1*, int *site2* )

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

##### Parameters

|              |                 |
|--------------|-----------------|
| <i>site1</i> | the first site  |
| <i>site2</i> | the second site |

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

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

##### Parameters

|              |                             |
|--------------|-----------------------------|
| <i>site1</i> | the first site              |
| <i>site2</i> | 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).

#### 7.14.4.41 int Langmuir::Grid::zImageDistancei ( int *site1*, int *site2* )

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

##### Parameters

|              |                             |
|--------------|-----------------------------|
| <i>site1</i> | the first site              |
| <i>site2</i> | 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).

#### 7.14.4.42 int Langmuir::Grid::zSize ( )

Get the number of sites along the z-direction.

### 7.14.5 Member Data Documentation

#### 7.14.5.1 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.

#### 7.14.5.2 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.

**7.14.5.3** `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.

**7.14.5.4** `int Langmuir::Grid::m_specialAgentCount` `[protected]`

The current number of special Agents registered with the [Grid](#).

**7.14.5.5** `int Langmuir::Grid::m_specialAgentReserve` `[protected]`

The max number of special Agents allowed.

**7.14.5.6** `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](#).

**7.14.5.7** `int Langmuir::Grid::m_volume` `[protected]`

The total number of sites.

**7.14.5.8** `World& Langmuir::Grid::m_world` `[protected]`

Reference to the [World](#) object.

**7.14.5.9** `int Langmuir::Grid::m_xSize` `[protected]`

The number of sites along the x-direction.

**7.14.5.10** `int Langmuir::Grid::m_xyPlaneArea` `[protected]`

The number of sites in the xy-plane.

**7.14.5.11** `int Langmuir::Grid::m_xzPlaneArea` `[protected]`

The number of sites in the xz-plane.

**7.14.5.12** `int Langmuir::Grid::m_ySize` `[protected]`

The number of sites along the y-direction.

#### 7.14.5.13 `int Langmuir::Grid::m_yzPlaneArea` [protected]

The number of sites in the yz-plane.

#### 7.14.5.14 `int Langmuir::Grid::m_zSize` [protected]

The number of sites along the z-direction.

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

- [cubicgrid.h](#)
- [cubicgrid.cpp](#)

## 7.15 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*

### 7.15.1 Detailed Description

A class to draw images of the grid.

### 7.15.2 Constructor & Destructor Documentation

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

create the image and painter, setting the background and size

#### Parameters

|               |                               |
|---------------|-------------------------------|
| <i>world</i>  | Reference to the world object |
| <i>bg</i>     | Background color              |
| <i>parent</i> | parent QObject                |

### 7.15.3 Member Function Documentation

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

draw some sites

##### Parameters

|                |  |
|----------------|--|
| <i>charges</i> | A list of ChargeAgents, which have site ids <ul style="list-style-type: none"> <li>• could be the list of electrons</li> <li>• could be the list of holes</li> </ul> |
| <i>color</i>   | The color of the points  |
| <i>layer</i>   | Which layer are we drawing? its a 2D image   |

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

draw some sites

##### Parameters

|              |   |
|--------------|---|
| <i>sites</i> | A list of integers that are site ids <ul style="list-style-type: none"> <li>• could be the list of trap ids</li> <li>• could be the list of defect ids</li> </ul> |
| <i>color</i> | The color of the points   |
| <i>layer</i> | Which layer are we drawing? its a 2D image  |

#### 7.15.3.3 void Langmuir::GridImage::save ( QString name, int scale = 3 )

save the image to a file

##### Parameters

|              |  |
|--------------|--|
| <i>name</i>  | A file name that is passed to a <a href="#">OutputInfo</a> object, the output is assumed png |
| <i>scale</i> | Multiply the image by some scale, increasing the resolution                                  |

### 7.15.4 Member Data Documentation

#### 7.15.4.1 QImage Langmuir::GridImage::m\_image [private]

the image we draw onto

#### 7.15.4.2 QPainter Langmuir::GridImage::mPainter [private]

the painter that paints the image

#### 7.15.4.3 World& Langmuir::GridImage::m\_world [private]

reference to the world object

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

- [writer.h](#)
- [writer.cpp](#)

## 7.16 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

#### 7.16.1 Detailed Description

A class to represent moving positive charges.

#### 7.16.2 Constructor & Destructor Documentation

7.16.2.1 [Langmuir::HoleAgent::HoleAgent](#) ( [World](#) & *world*, int *site*, QObject \* *parent* = 0 )

Construct [HoleAgent](#).

#### 7.16.3 Member Function Documentation

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

Calculate Exciton Binding Energy.

##### Parameters

|             |   |
|-------------|---|
| <i>site</i> | the site to check in other <a href="#">Grid</a> |
|-------------|---|

##### Returns

- −0.5 eV if exciton
- 0 otherwise

Implements [Langmuir::ChargeAgent](#).

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

Return other [Grid](#).

Returns

[World::electronGrid](#)

Implements [Langmuir::ChargeAgent](#).

### 7.16.3.3 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 files:

- [chargeagent.h](#)
- [chargeagent.cpp](#)

## 7.17 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

#### 7.17.1 Detailed Description

A class to remove HoleAgents.

#### 7.17.2 Constructor & Destructor Documentation

##### 7.17.2.1 Langmuir::HoleDrainAgent::HoleDrainAgent ( [World](#) & world, int site, QObject \* parent = 0 )

create an [HoleDrainAgent](#) at a specific site

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

create a [HoleDrainAgent](#) at a specific [Grid::CubeFace](#)

### 7.17.3 Member Function Documentation

7.17.3.1 `double Langmuir::HoleDrainAgent::energyChange ( int fSite )` `[protected]`, `[virtual]`

same as [FluxAgent::energyChange\(\)](#), but specialized for HoleAgents.

Note really used because the default [FluxAgent::shouldTransport\(\)](#) behavior, 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 files:

- [drainagent.h](#)
- [drainagent.cpp](#)

## 7.18 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

#### 7.18.1 Detailed Description

A class to inject HoleAgents.

#### 7.18.2 Constructor & Destructor Documentation

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

create a [HoleSourceAgent](#) at a specific site



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

create a [HoleSourceAgent](#) at a specific [Grid::CubeFace](#)

### 7.18.3 Member Function Documentation

7.18.3.1 `double Langmuir::HoleSourceAgent::energyChange ( int site )` `[protected]`, `[virtual]`

same as [FluxAgent::energyChange\(\)](#), but specialized for HoleAgents.

Reimplemented from [Langmuir::FluxAgent](#).

7.18.3.2 `void Langmuir::HoleSourceAgent::inject ( int site )` `[protected]`, `[virtual]`

same as [SourceAgent::inject\(\)](#), but specialized for HoleAgents.

Implements [Langmuir::SourceAgent](#).

7.18.3.3 `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 files:

- [sourceagent.h](#)
- [sourceagent.cpp](#)

## 7.19 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.*

## 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.*

### 7.19.1 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](#).

### 7.19.2 Constructor & Destructor Documentation

#### 7.19.2.1 `Langmuir::KeyValueParser::KeyValueParser ( World & world, QObject * parent = 0 )`

Create a [KeyValueParser](#).

#### Parameters

|               |   |
|---------------|---|
| <i>world</i>  | reference to <a href="#">World</a> Object |
| <i>parent</i> | QObject this belongs to                   |

Add calls to [registerVariable\(\)](#) to add new variables to the simulation.

## 7.19.2.2 Langmuir::KeyValueParser::~~KeyValueParser ( )

Destroy the [KeyValueParser](#).

## 7.19.3 Member Function Documentation

## 7.19.3.1 Variable &amp; Langmuir::KeyValueParser::getVariable ( const QString &amp; key )

Get a reference to a variable by name.

## Parameters

|             |                          |
|-------------|--------------------------|
| <i>name</i> | the name of the variable |
|-------------|--------------------------|

## 7.19.3.2 SimulationParameters &amp; Langmuir::KeyValueParser::parameters ( )

Get the [SimulationParameters](#).

## 7.19.3.3 void Langmuir::KeyValueParser::parse ( const QString &amp; line )

Parse a string and assign a value to the correct parameter.

## 7.19.3.4 template&lt;typename T &gt; void Langmuir::KeyValueParser::registerVariable ( const QString &amp; key, T &amp; value, Variable::VariableMode mode = 0 ) [private]

Register an allowed variable with the parser.

## 7.19.3.5 void Langmuir::KeyValueParser::save ( const QString &amp; fileName = "%stub.parm" )

Write the parameters to a file in a "key=value" fashion.

## 7.19.4 Friends And Related Function Documentation

## 7.19.4.1 std::ostream&amp; operator&lt;&lt; ( std::ostream &amp; stream, const KeyValueParser &amp; keyValueParser ) [friend]

Write the parameters to a std::ostream in a "key=value" fashion.

## 7.19.5 Member Data Documentation

## 7.19.5.1 QStringList Langmuir::KeyValueParser::m\_orderedNames [private]

A list of variable names in a specific order.

## 7.19.5.2 SimulationParameters Langmuir::KeyValueParser::m\_parameters [private]

The simulation parameters.

## 7.19.5.3 QMap&lt;QString,Variable\*&gt; Langmuir::KeyValueParser::m\_variableMap [private]

A map between variable names and variables.

#### 7.19.5.4 World & Langmuir::KeyValueParser::m\_world [private]

Reference to [World](#) object.

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

- [keyvalueparser.h](#)
- [keyvalueparser.cpp](#)

## 7.20 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)*

### 7.20.1 Detailed Description

A class that organizes output.

#### Warning

You must manually call [initialize\(\)](#) to open output streams

### 7.20.2 Constructor & Destructor Documentation

7.20.2.1 `Langmuir::Logger::Logger ( World & world, QObject * parent = 0 )`

create [Logger](#)

### 7.20.3 Member Function Documentation

7.20.3.1 `void Langmuir::Logger::initialize ( )` [[virtual](#)]

open the various output streams if they are turned on

7.20.3.2 `void Langmuir::Logger::reportCarrier ( ChargeAgent & charge )` [[virtual](#)]

output carrier information (for example pathlength) to the carrier file

7.20.3.3 `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

7.20.3.4 `void Langmuir::Logger::reportFluxStream ( )` [[virtual](#)]

output information about Sources and Drains (at the current step) to the main output file

7.20.3.5 `void Langmuir::Logger::reportXYZStream ( )` [[virtual](#)]

output xyz information (at the current step) to the xyz file

**7.20.3.6** `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

**7.20.3.7** `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**

**7.20.3.8** `void Langmuir::Logger::saveDefectImage ( const QString & name = "%stub-defects.png" )`  
[virtual]

save an image of defects as png

**7.20.3.9** `void Langmuir::Logger::saveElectronImage ( const QString & name = "%stub-%step-electrons.png" )`  
[virtual]

save an image of electrons (at the current step) as png

**7.20.3.10** `void Langmuir::Logger::saveGridPotential ( const QString & name = "%stub.grid" )` [virtual]

output the grid potential as (x, y, z, v) to a file

**7.20.3.11** `void Langmuir::Logger::saveHoleImage ( const QString & name = "%stub-%step-holes.png" )`  
[virtual]

save an image of holes (at the current step) as png

**7.20.3.12** `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

**7.20.3.13** `void Langmuir::Logger::saveTrapImage ( const QString & name = "%stub-traps.png" )` [virtual]

save an image of trap sites as png

## 7.20.4 Member Data Documentation

**7.20.4.1** `CarrierWriter* Langmuir::Logger::m_carrierWriter` [protected]

writer in charge of writing carrier information

**7.20.4.2** `ExcitonWriter* Langmuir::Logger::m_excitonWriter` [protected]

writer in charge of writing multiple carrier's information (excitons)

**7.20.4.3** `FluxWriter* Langmuir::Logger::m_fluxWriter` [protected]

writer in charge of writing source & drain information

## 7.20.4.4 World &amp; Langmuir::Logger::m\_world [protected]

reference to world

## 7.20.4.5 XYZWriter\* Langmuir::Logger::m\_xyzWriter [protected]

writer in charge of writing xyz files

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

- [writer.h](#)
- [writer.cpp](#)

## 7.21 Langmuir::OpenCIHelper Class Reference

A Class to run OpenCL calculations.

```
#include <openclhelper.h>
```

### Public Member Functions

- [OpenCIHelper](#) ([World](#) &world, QObject \*parent=0)  
*Create **THE** [OpenCIHelper](#); don't make more than one.*
- void [initializeOpenCL](#) ()  
*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.*

### 7.21.1 Detailed Description

A Class to run OpenCL calculations.

### 7.21.2 Constructor & Destructor Documentation

#### 7.21.2.1 `Langmuir::OpenCIHelper::OpenCIHelper ( World & world, QObject * parent = 0 )`

Create **THE** [OpenCIHelper](#); don't make more than one.

##### Parameters

|               |   |
|---------------|---|
| <i>world</i>  | reference to <a href="#">World</a> Object |
| <i>parent</i> | QObject this belongs to                   |

##### Warning

[initializeOpenCL\(\)](#) must be called separately

### 7.21.3 Member Function Documentation

#### 7.21.3.1 `void Langmuir::OpenCIHelper::compareHostAndDeviceForAllCarriers ( )`

Compare GPU and CPU results.

#### 7.21.3.2 `void Langmuir::OpenCIHelper::copySiteAndChargeToHostVector ( int index, int site, int charge = -1 )`

Does exactly what it says (host means the memory on the CPU)

##### Parameters

|               |                          |
|---------------|--------------------------|
| <i>index</i>  | position in host vectors |
| <i>site</i>   | serial site-id           |
| <i>charge</i> | charge of carrier        |

#### 7.21.3.3 `double Langmuir::OpenCIHelper::getOutputHost ( int index ) const`

Get the result stored in host memory (for current site)

##### Parameters

|              |                          |
|--------------|--------------------------|
| <i>index</i> | position in host vectors |
|--------------|--------------------------|

#### 7.21.3.4 `double Langmuir::OpenCIHelper::getOutputHostFuture ( int index ) const`

Get the result stored in host memory (for future site)

##### Parameters

|              |                          |
|--------------|--------------------------|
| <i>index</i> | position in host vectors |
|--------------|--------------------------|

There is a fixed offset in the host memory between the current and future site results



## 7.21.3.5 void Langmuir::OpenCIHelper::initializeOpenCL ( )

Perform the tedious boilerplate code to initialize OpenCL.

## 7.21.3.6 void Langmuir::OpenCIHelper::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.

## 7.21.3.7 void Langmuir::OpenCIHelper::launchCoulombKernel2 ( )

Kernel2 calculates the coulomb potential at current and future sites only.

## 7.21.3.8 void Langmuir::OpenCIHelper::launchGaussKernel1 ( )

Kernel1 calculates the coulomb potential with erf at **every** site.

## 7.21.3.9 void Langmuir::OpenCIHelper::launchGaussKernel2 ( )

Kernel2 calculates the coulomb potential with erf at current and future sites only.

7.21.3.10 bool Langmuir::OpenCIHelper::toggleOpenCL ( bool *on* )

Turn on/off OpenCL in a smart-way.

## Parameters

|           |            |
|-----------|------------|
| <i>on</i> | 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.

## 7.21.4 Member Data Documentation

## 7.21.4.1 World&amp; Langmuir::OpenCIHelper::m\_world [private]

Reference to [World](#) object.

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

- [opencihelper.h](#)
- [opencihelper.cpp](#)

## 7.22 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](#).*

### 7.22.1 Detailed Description

A class to generate file names using the [SimulationParameters](#).

### 7.22.2 Constructor & Destructor Documentation

#### 7.22.2.1 `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

|             |   |
|-------------|---|
| <i>name</i> | the file name desired. The following substitutions can be made: <ul style="list-style-type: none"> <li>• <code>"%stub"</code>, substitutes in <a href="#">SimulationParameters::outputStub</a></li> <li>• <code>"%step"</code>, substitutes in <a href="#">SimulationParameters::currentStep</a></li> </ul> |
| <i>par</i>  | pointer to a <a href="#">SimulationParameters</a> object <ul style="list-style-type: none"> <li>• if 0 or NULL, then all substitutions become empty strings</li> </ul>  |

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

- [output.h](#)
- [output.cpp](#)

## 7.23 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.*

### 7.23.1 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.

### 7.23.2 Constructor & Destructor Documentation

**7.23.2.1** `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](#)

**7.23.2.2** `Langmuir::OutputStream::~~OutputStream ( )`

Flush the stream and close the file.

### 7.23.3 Member Function Documentation

**7.23.3.1** `const QFile & Langmuir::OutputStream::file ( )`

Get the file object, though you probably have no need for it.

**7.23.3.2** `const OutputInfo & Langmuir::OutputStream::info ( )`

Get the info object to get things like file name and path.

### 7.23.4 Member Data Documentation

**7.23.4.1** `QFile Langmuir::OutputStream::m_file` [private]

QFile object, the device of this QTextStream.

**7.23.4.2** `OutputInfo Langmuir::OutputStream::m_info` [private]

[OutputInfo](#) object that generated file name.

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

- [output.h](#)
- [output.cpp](#)

## 7.24 Langmuir::PBSGPUParser Class Reference

```
#include <pbsgpuparser.h>
```

### Public Member Functions

- [PBSGPUParser](#) (QObject \*parent=0)  
*create the [PBSGPUParser](#)*
- void [setPath](#) (const QString &path="")  
*set the path of the PBS\_GPUFILE*
- void [acquirePath](#) ()  
*get the path of the PBS\_GPUFILE from the ENVIRONMENT variables*
- void [setDefault](#) ()  
*reset to default GPU(0)*
- void [parse](#) ()  
*parse the PBS\_GPUFILE*
- QList< int > [nodes](#) ()  
*get the list of nodes ids found*
- QList< int > [gpus](#) ()  
*get the list of GPU ids found*
- int [size](#) ()  
*get the number of GPUs found*
- int [node](#) ()  
*get the first node found*
- int [gpu](#) ()  
*get the first GPU found*

### Private Attributes

- QList< int > [m\\_nodes](#)  
*list of nodes*
- QList< int > [m\\_gpus](#)  
*list of GPU ids*
- QString [m\\_path](#)  
*path to PBS\_GPUFILE*

#### 7.24.1 Detailed Description

A class to parse the PBS\_GPUFILE - needed for selecting the correct GPU on a cluster using PBS

#### 7.24.2 Constructor & Destructor Documentation

7.24.2.1 Langmuir::PBSGPUParser::PBSGPUParser ( QObject \* *parent* = 0 ) [\[explicit\]](#)

create the [PBSGPUParser](#)

### 7.24.3 Member Function Documentation

#### 7.24.3.1 void Langmuir::PBSGPUParser::acquirePath ( )

get the path of the PBS\_GPUFILE from the ENVIRONMENT variables

#### 7.24.3.2 int Langmuir::PBSGPUParser::gpu ( )

get the first GPU found

#### 7.24.3.3 QList< int > Langmuir::PBSGPUParser::gpus ( )

get the list of GPU ids found

#### 7.24.3.4 int Langmuir::PBSGPUParser::node ( )

get the first node found

#### 7.24.3.5 QList< int > Langmuir::PBSGPUParser::nodes ( )

get the list of nodes ids found

#### 7.24.3.6 void Langmuir::PBSGPUParser::parse ( )

parse the PBS\_GPUFILE

#### 7.24.3.7 void Langmuir::PBSGPUParser::setDefault ( )

reset to default GPU(0)

#### 7.24.3.8 void Langmuir::PBSGPUParser::setPath ( const QString & *path* = " " )

set the path of the PBS\_GPUFILE

#### 7.24.3.9 int Langmuir::PBSGPUParser::size ( )

get the number of GPUs found

### 7.24.4 Member Data Documentation

#### 7.24.4.1 QList<int> Langmuir::PBSGPUParser::m\_gpus [private]

list of GPU ids

#### 7.24.4.2 QList<int> Langmuir::PBSGPUParser::m\_nodes [private]

list of nodes

#### 7.24.4.3 QString Langmuir::PBSGPUParser::m\_path [private]

path to PBS\_GPUFILE

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

- [pbsgpuparser.h](#)
- [pbsgpuparser.cpp](#)

## 7.25 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 calculated 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](#)*

## 7.25.1 Detailed Description

A class to calculate the potential.

## 7.25.2 Constructor & Destructor Documentation

7.25.2.1 [Langmuir::Potential::Potential](#) ( [World](#) & *world*, [QObject](#) \* *parent* = 0 )

[Potential](#) Create the potential.

### Parameters

|               |   |
|---------------|---|
| <i>world</i>  | reference to the <a href="#">World</a>  |
| <i>parent</i> | <a href="#">QObject</a> this belongs to |

## 7.25.3 Member Function Documentation

7.25.3.1 [double Langmuir::Potential::coulombD](#) ( int *site.i* )

calculates Coulomb potential from charged defects at specific grid site

### Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

7.25.3.2 [double Langmuir::Potential::coulombE](#) ( int *site.i* )

calculates Coulomb potential from electrons at specific grid site

### Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

7.25.3.3 [double Langmuir::Potential::coulombH](#) ( int *site.i* )

calculates Coulomb potential from holes at specific grid site

### Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.4 double Langmuir::Potential::coulombImageD ( int *site.i* )**

calculates Coulomb image-potential from charged defects at specific grid site

**Parameters**

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.5 double Langmuir::Potential::coulombImageE ( int *site.i* )**

calculates Coulomb image-potential from electrons at specific grid site

**Parameters**

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.6 double Langmuir::Potential::coulombImageH ( int *site.i* )**

calculates Coulomb image-potential from holes at specific grid site

**Parameters**

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.7 double Langmuir::Potential::gaussD ( int *site.i* )**

calculates Coulomb potential from charged defects at specific grid site, assuming gaussians

**Parameters**

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.8 double Langmuir::Potential::gaussE ( int *site.i* )**

calculates Coulomb potential from electrons at specific grid site, assuming gaussians

**Parameters**

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.9 double Langmuir::Potential::gaussH ( int *site* )**

calculates Coulomb potential from holes at specific grid site, assuming gaussians

**Parameters**

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

**7.25.3.10 double Langmuir::Potential::gaussImageD ( int *site.i* )**

calculates Coulomb image-potential from charged defects at specific grid site, assuming gaussians



## Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

7.25.3.11 double Langmuir::Potential::gaussImageE ( int *site.i* )

calculates Coulomb image-potential from electrons at specific grid site, assuming gaussians

## Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

7.25.3.12 double Langmuir::Potential::gaussImageH ( int *site* )

calculates Coulomb image-potential from holes at specific grid site, assuming gaussians

## Parameters

|             |                      |
|-------------|----------------------|
| <i>site</i> | the site of interest |
|-------------|----------------------|

## 7.25.3.13 void Langmuir::Potential::precalculateArrays ( )

pre-calculates r2, r, and 1/r

## 7.25.3.14 void Langmuir::Potential::setPotentialGate ( )

Adds a linear potential calculated from slope.z along the z-direction.

## 7.25.3.15 void Langmuir::Potential::setPotentialLinear ( )

Adds a linear potential calculated from voltage.left and voltage.right along the x-direction.

7.25.3.16 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

|                       |                  |
|-----------------------|------------------|
| <i>trapIDs</i>        | list of site ids |
| <i>trapPotentials</i> | list of shifts   |

## 7.25.3.17 void Langmuir::Potential::setPotentialZero ( )

sets the value of the potential to zero at every grid site

## 7.25.3.18 void Langmuir::Potential::updateCouplingConstants ( )

pre-calculates coupling constants

## 7.25.4 Member Data Documentation

### 7.25.4.1 World& Langmuir::Potential::m\_world [private]

reference to the [World](#)

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

- [potential.h](#)
- [potential.cpp](#)

## 7.26 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`.

### 7.26.1 Detailed Description

A class to generate random numbers.

### 7.26.2 Constructor & Destructor Documentation

7.26.2.1 `Langmuir::Random::Random ( quint64 seed = 0, QObject * parent = 0 )`

[Random](#).

#### Parameters

|               |  |
|---------------|--|
| <i>seed</i>   | makes the generator deterministic <ul style="list-style-type: none"> <li>• <code>seed == 0</code> uses the current clock time</li> </ul> |
| <i>parent</i> | object this belongs to   |

7.26.2.2 `Langmuir::Random::~~Random ( )`

Destroy objects.

### 7.26.3 Member Function Documentation

7.26.3.1 `bool Langmuir::Random::chooseNo ( double percent )`

Randomly choose no a percent of the time.

7.26.3.2 `bool Langmuir::Random::chooseYes ( double percent )`

Randomly choose yes a percent of the time.

7.26.3.3 `int Langmuir::Random::integer ( const int low = 0, const int high = 1 )`

Generate a random int from the uniform distribution [low, high].

#### 7.26.3.4 `bool Langmuir::Random::metropolis ( double energyChange, double inversekT )`

Randomly choose yes using a Boltzmann factor.

##### Parameters

|                     |   |
|---------------------|---|
| <i>energyChange</i> | change in energy when going from initial to final state |
| <i>inversekT</i>    | decay constant in exponential                           |

#### 7.26.3.5 `bool Langmuir::Random::metropolisWithCoupling ( double energyChange, double inversekT, double coupling )`

Randomly choose yes using a Boltzmann factor and coupling constant.

##### Parameters

|                     |   |
|---------------------|---|
| <i>energyChange</i> | change in energy when going from initial to final state   |
| <i>inversekT</i>    | decay constant in exponential   |
| <i>coupling</i>     | alters acceptance probability <ul style="list-style-type: none"> <li>• if energy &lt; 0 : chooses yes coupling * Boltzmann factor percent of the time</li> <li>• if energy &gt; 0 : chooses yes 1 - coupling percent of the time</li> </ul> |

#### 7.26.3.6 `double Langmuir::Random::normal ( const double mean, const double sigma )`

Generate a random double from the normal distribution.

##### Parameters

|              |   |
|--------------|---|
| <i>mean</i>  | average value of the normal distribution sampled      |
| <i>sigma</i> | standard deviation of the normal distribution sampled |

#### 7.26.3.7 `double Langmuir::Random::random ( )`

Generate a random double from the uniform distribution [0, 1].

#### 7.26.3.8 `double Langmuir::Random::range ( const double low = 0.0, const double high = 1.0 )`

Generate a random double from the uniform distribution [low, high].

#### 7.26.3.9 `quint64 Langmuir::Random::seed ( )`

Get the seed that was used.

#### 7.26.3.10 `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](#).

## 7.26.4 Friends And Related Function Documentation

7.26.4.1 `QDataStream& operator<< ( QDataStream & stream, Random & random )` `[friend]`

Output the random state to a QDataStream **Possibly Broken**.

#### Warning

This may not quite be working correctly

7.26.4.2 `QTextStream& operator<< ( QTextStream & stream, Random & random )` `[friend]`

Output the random state to a QTextStream **Possibly Broken**.

#### Warning

This may not quite be working correctly

7.26.4.3 `std::ostream& operator<< ( std::ostream & stream, Random & random )` `[friend]`

Output the random state to a std::ostream.

7.26.4.4 `QDataStream& operator>> ( QDataStream & stream, Random & random )` `[friend]`

Load the random state from a QDataStream **Possibly Broken**.

#### Warning

This may not quite be working correctly

7.26.4.5 `QTextStream& operator>> ( QTextStream & stream, Random & random )` `[friend]`

Load the random state from a QTextStream **Possibly Broken**.

#### Warning

This may not quite be working correctly

7.26.4.6 `std::istream& operator>> ( std::istream & stream, Random & random )` `[friend]`

Load the random state from a std::istream.

## 7.26.5 Member Data Documentation

7.26.5.1 `boost::variate_generator<boost::mt19937&, boost::uniform_01<double> >* Langmuir::Random::generator01`  
`[private]`

The underlying generator coupled to the uniform distribution on [0,1].

7.26.5.2 `quint64 Langmuir::Random::m_seed` `[private]`

The seed used to start the generator.

### 7.26.5.3 boost::mt19937\* Langmuir::Random::twister [private]

The underlying random number generator.

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

- [rand.h](#)
- [rand.cpp](#)

## 7.27 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

#### 7.27.1 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.

#### 7.27.2 Constructor & Destructor Documentation

##### 7.27.2.1 Langmuir::RecombinationAgent::RecombinationAgent ( [World](#) & world, [QObject](#) \* parent = 0 )

create a [RecombinationAgent](#)

#### 7.27.3 Member Function Documentation

##### 7.27.3.1 double Langmuir::RecombinationAgent::energyChange ( int fSite ) [private],[virtual]

currently implemented as zero and not really used

## 7.27.3.2 void Langmuir::RecombinationAgent::guessProbability ( )

calculate an acceptance probability based upon the desired rate and encounter frequency

## 7.27.3.3 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 files:

- [drainagent.h](#)
- [drainagent.cpp](#)

## 7.28 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.*

### 7.28.1 Detailed Description

A class to orchestrate the calculation.

### 7.28.2 Constructor & Destructor Documentation

#### 7.28.2.1 `Langmuir::Simulation::Simulation ( World & world, QObject * parent = 0 )`

Create a [Simulation](#).

Parameters

|               |   |
|---------------|---|
| <i>world</i>  | reference to <a href="#">World</a> Object |
| <i>parent</i> | QObject this belongs to                   |

#### 7.28.2.2 `Langmuir::Simulation::~~Simulation ( ) [virtual]`

Destroy the [Simulation](#).

### 7.28.3 Member Function Documentation

#### 7.28.3.1 `void Langmuir::Simulation::balanceCharges ( ) [protected]`

**Try** to use the sources to keep the number of ChargeAgents balanced

#### 7.28.3.2 `void Langmuir::Simulation::chargeAgentCoulombInteractionQtConcurrentCPU ( ChargeAgent * chargeAgent ) [inline], [static], [protected]`

A method needed to call [ChargeAgent::coulombCPU\(\)](#) in parallel.

#### 7.28.3.3 `void Langmuir::Simulation::chargeAgentCoulombInteractionQtConcurrentGPU ( ChargeAgent * chargeAgent ) [inline], [static], [protected]`

A method needed to call [ChargeAgent::coulombGPU\(\)](#) in parallel.

Does not perform GPU calculations. 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.

#### 7.28.3.4 `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.

#### 7.28.3.5 `void Langmuir::Simulation::performInjections ( ) [protected]`

Tell sources to inject charges.

#### 7.28.3.6 `void Langmuir::Simulation::performIterations ( int nIterations ) [virtual]`

simulate for a set number of steps



## Parameters

|                    |                                 |
|--------------------|---------------------------------|
| <i>nIterations</i> | the number of steps to simulate |
|--------------------|---------------------------------|

7.28.3.7 void Langmuir::Simulation::performRecombinations ( ) [protected]

Recombine holes and electrons (in solarcell simulations only)

## 7.28.4 Member Data Documentation

7.28.4.1 World& Langmuir::Simulation::m\_world [protected]

Reference to [World](#) object.

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

- [simulation.h](#)
- [simulation.cpp](#)

## 7.29 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  $n * \text{iterations.print steps}$ )*
- 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  $n * \text{iterations.print steps}$ )*
- 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*
- quint32 [workX](#)  
*the x size of OpenCL 3DRange kernel work groups - only needed if using [SimulationParameters::outputCoulomb](#)*
- quint32 [workY](#)  
*the y size of OpenCL 3DRange kernel work groups - only needed if using [SimulationParameters::outputCoulomb](#)*
- quint32 [workZ](#)  
*the z size of OpenCL 3DRange kernel work groups - only needed if using [SimulationParameters::outputCoulomb](#)*
- quint32 [workSize](#)  
*the size of OpenCL 1DRange kernel work groups*
- quint32 [opencLThreshold](#)  
*the minimum number of charges that must be present to use OpenCL*
- quint32 [opencLDeviceID](#)  
*the device to choose if there are multiple*
- qreal [boltzmannConstant](#)  
*physical constant, the boltzmann constant*
- qreal [dielectricConstant](#)

- physical constant, the dielectric constant*
- qreal [elementaryCharge](#)
  - physical constant, the elementary charge*
- qreal [permittivitySpace](#)
  - physical constant, the permittivity of free space*
- qreal [gridFactor](#)
  - size constant, the size associated with grid sites (~ 1nm)*
- quint32 [electrostaticCutoff](#)
  - the cut off for Coulomb interactions*
- 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*
- quint32 [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 calculating energy change for injection*
- qreal [recombinationRate](#)
  - the rate at which holes and electrons can combine when they sit upon one another*
- quint32 [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::xy-PlaneArea)/(SimulationParameters::sourceScaleArea)`; if `<= 0`, does not scale rate*
- quint32 [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*

### 7.29.1 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](#).

## 7.29.2 Constructor & Destructor Documentation

7.29.2.1 `Langmuir::SimulationParameters::SimulationParameters ( )` `[inline]`

## 7.29.3 Member Data Documentation

7.29.3.1 `bool Langmuir::SimulationParameters::balanceCharges`

if true, try to keep the number of charges in the simulation balanced

7.29.3.2 `qreal Langmuir::SimulationParameters::boltzmannConstant`

physical constant, the boltzmann constant

7.29.3.3 `bool Langmuir::SimulationParameters::coulombCarriers`

turn on Coulomb interactions between ChargeAgents

7.29.3.4 `qreal Langmuir::SimulationParameters::coulombGaussianSigma`

multiply Coulomb terms by  $\text{erf}[r/(\text{sigma} \sqrt{2})]$ ; nothing happens if its zero

7.29.3.5 `quint32 Langmuir::SimulationParameters::currentStep`

the current step of the simulation

7.29.3.6 `qreal Langmuir::SimulationParameters::defectPercentage`

the percent of the grid that is reserved for electrons, between 0 and 1

7.29.3.7 `quint32 Langmuir::SimulationParameters::defectsCharge`

the charge of defect sites

7.29.3.8 `qreal Langmuir::SimulationParameters::dielectricConstant`

physical constant, the dielectric constant

7.29.3.9 `qreal Langmuir::SimulationParameters::drainRate`

the rate at which all drains accept charges (default, used when `eDrainL`, etc. are  $< 0$ )

7.29.3.10 `qreal Langmuir::SimulationParameters::eDrainLRate`

the rate at which the left electron drain accepts charges (overrides default)

7.29.3.11 `qreal Langmuir::SimulationParameters::eDrainRRate`

the rate at which the right electron drain accepts charges (overrides default)

#### 7.29.3.12 `qreal Langmuir::SimulationParameters::electronPercentage`

the percent of the grid that is reserved for electrons, between 0 and 1

#### 7.29.3.13 `qint32 Langmuir::SimulationParameters::electrostaticCutoff`

the cut off for Coulomb interactions

#### 7.29.3.14 `qreal Langmuir::SimulationParameters::electrostaticPrefactor`

a compilation of physical constants

#### 7.29.3.15 `qreal Langmuir::SimulationParameters::elementaryCharge`

physical constant, the elementary charge

#### 7.29.3.16 `qreal Langmuir::SimulationParameters::eSourceLRate`

the rate at which the left electron source injects charges (overrides default)

#### 7.29.3.17 `qreal Langmuir::SimulationParameters::eSourceRRate`

the rate at which the right electron source injects charges (overrides default)

#### 7.29.3.18 `qreal Langmuir::SimulationParameters::excitonBinding`

the energy change (eV) when a hole/electron pair goes from the same site to adjacent sites

#### 7.29.3.19 `qreal Langmuir::SimulationParameters::gaussianStdev`

the standard deviation of trap sites

#### 7.29.3.20 `qreal Langmuir::SimulationParameters::generationRate`

the rate at which the exciton source injects charges (overrides default)

#### 7.29.3.21 `qreal Langmuir::SimulationParameters::gridFactor`

size constant, the size associated with grid sites ( $\sim 1\text{nm}$ )

#### 7.29.3.22 `qint32 Langmuir::SimulationParameters::gridX`

the number of sites along the device length, at least one

#### 7.29.3.23 `qint32 Langmuir::SimulationParameters::gridY`

the number of sites along the device width, at least one

**7.29.3.24   qint32 Langmuir::SimulationParameters::gridZ**

the number of sites per layer, at least one

**7.29.3.25   qreal Langmuir::SimulationParameters::hDrainLRate**

the rate at which the left hole drain accepts charges (overrides default)

**7.29.3.26   qreal Langmuir::SimulationParameters::hDrainRRate**

the rate at which the right hole drain accepts charges (overrides default)

**7.29.3.27   qreal Langmuir::SimulationParameters::holePercentage**

the percent of the grid that is reserved for holes, between 0 and 1

**7.29.3.28   qint32 Langmuir::SimulationParameters::hoppingRange**

the number of sites away from a given site used when calculating neighboring sites

**7.29.3.29   qreal Langmuir::SimulationParameters::hSourceLRate**

the rate at which the left hole source injects charges (overrides default)

**7.29.3.30   qreal Langmuir::SimulationParameters::hSourceRRate**

the rate at which the right hole source injects charges (overrides default)

**7.29.3.31   qint32 Langmuir::SimulationParameters::imageCarriers**

output images of carriers (if  $n < 0$ , only at the end; if  $n == 0$ , never; if  $n > 0$ , every  $n * \text{iterations.print steps}$ )

**7.29.3.32   bool Langmuir::SimulationParameters::imageDefects**

output images of defects

**7.29.3.33   bool Langmuir::SimulationParameters::imageTraps**

output images of defects

**7.29.3.34   qreal Langmuir::SimulationParameters::inverseKT**

a compilation of physical constants

**7.29.3.35   qint32 Langmuir::SimulationParameters::iterationsPrint**

if [Langmuir](#), how often to output; if [LangmuirView](#), how many steps between rendering

**7.29.3.36   qint32 Langmuir::SimulationParameters::iterationsReal**

number of simulation steps after equilibration

**7.29.3.37   qint32 Langmuir::SimulationParameters::maxThreads**

max threads allowed for QThreadPool - if its  $\leq 0$  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

**7.29.3.38   bool Langmuir::SimulationParameters::okCL**

if true, OpenCL can be used on this platform

**7.29.3.39   qint32 Langmuir::SimulationParameters::opencldDeviceID**

the device to choose if there are multiple

**7.29.3.40   qint32 Langmuir::SimulationParameters::opencldThreshold**

the minimum number of charges that must be present to use OpenCL

**7.29.3.41   bool Langmuir::SimulationParameters::outputChkTrapPotential**

output trap potentials in checkpoint files

**7.29.3.42   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  $n * \text{iterations.print steps}$ )

**7.29.3.43   bool Langmuir::SimulationParameters::outputIdsOnDelete**

output carrier lifetime and pathlength when they are deleted

**7.29.3.44   bool Langmuir::SimulationParameters::outputIdsOnEncounter**

output carrier lifetime and pathlength when holes and electrons encounter one another

**7.29.3.45   bool Langmuir::SimulationParameters::outputIsOn**

if false, produce no output (useful for LangmuirView)

**7.29.3.46   bool Langmuir::SimulationParameters::outputPotential**

output grid potential at the start of the simulation, includes the trap potential

**7.29.3.47   qint32 Langmuir::SimulationParameters::outputPrecision**

number of significant figures used for doubles in output



**7.29.3.48 qint32 Langmuir::SimulationParameters::outputStepChk**

output a checkpoint file every this \* iterationsPrint

**7.29.3.49 QString Langmuir::SimulationParameters::outputStub**

the stub to use when naming output files

**7.29.3.50 qint32 Langmuir::SimulationParameters::outputWidth**

width of columns in output, ignored in certain files, like trajectory files

**7.29.3.51 qint32 Langmuir::SimulationParameters::outputXyz**

output trajectory file (if  $n < 0$ , only at the end; if  $n == 0$ , never; if  $n > 0$ , every  $n * \text{iterations.print steps}$ )

**7.29.3.52 bool Langmuir::SimulationParameters::outputXyzD**

output defects in trajectory file (ignored if outputXyz is off)

**7.29.3.53 bool Langmuir::SimulationParameters::outputXyzE**

output electrons in trajectory file (ignored if outputXyz is off)

**7.29.3.54 bool Langmuir::SimulationParameters::outputXyzH**

output holes in trajectory file (ignored if outputXyz is off)

**7.29.3.55 qint32 Langmuir::SimulationParameters::outputXyzMode**

output mode for xyz file (if 0, particle count varies; if 1, particle count is constant using "phantom particles")

**7.29.3.56 bool Langmuir::SimulationParameters::outputXyzT**

output traps in trajectory file (ignored if outputXyz is off)

**7.29.3.57 qreal Langmuir::SimulationParameters::permittivitySpace**

physical constant, the permittivity of free spece

**7.29.3.58 quint64 Langmuir::SimulationParameters::randomSeed**

seed the random number generator, if negative, uses the current time (making seperate runs random)

**7.29.3.59 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)

**7.29.3.60   qreal Langmuir::SimulationParameters::recombinationRate**

the rate at which holes and electrons can combine when they sit upon one another

**7.29.3.61   qreal Langmuir::SimulationParameters::seedCharges**

if true, place charges randomly before the simulation starts

**7.29.3.62   qreal Langmuir::SimulationParameters::seedPercentage**

the percent of the traps to be placed and grown upon to form islands

**7.29.3.63   QDateTime Langmuir::SimulationParameters::simulationStart**

the time this simulation started

**7.29.3.64   QString Langmuir::SimulationParameters::simulationType**

tells [Langmuir](#) how to set up the Sources and Drains: ( "transistor", "solarcell" )

**7.29.3.65   qreal Langmuir::SimulationParameters::slopeZ**

slope of potential along z direction when there are multiple layers (as if there were a gate electrode)

**7.29.3.66   bool Langmuir::SimulationParameters::sourceCoulomb**

if true, use the Coulomb + Image interaction when calculating energy change for injection

**7.29.3.67   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)

**7.29.3.68   qreal Langmuir::SimulationParameters::sourceRate**

the rate at which all sources inject charges

**7.29.3.69   qreal Langmuir::SimulationParameters::sourceScaleArea**

for [SimulationParameters::simulationType](#) == "solarcell", multiply [SimulationParameters::sourceRate](#) by ([Grid::xy-PlaneArea](#))/([SimulationParameters::sourceScaleArea](#)); if <= 0, does not scale rate

**7.29.3.70   qreal Langmuir::SimulationParameters::temperatureKelvin**

the temperature used in the boltzmann factor

**7.29.3.71   qreal Langmuir::SimulationParameters::trapPercentage**

the percent of the grid that is reserved for traps, between 0 and 1

## 7.29.3.72 qreal Langmuir::SimulationParameters::trapPotential

the potential of traps

## 7.29.3.73 bool Langmuir::SimulationParameters::useOpenCL

if true, try to use OpenCL to speed up Coulomb interaction calculations

## 7.29.3.74 qreal Langmuir::SimulationParameters::voltageLeft

the potential on the left side of the grid, used in setting up an electric field

## 7.29.3.75 qreal Langmuir::SimulationParameters::voltageRight

the potential on the right side of the grid, used in setting up an electric field

## 7.29.3.76 qint32 Langmuir::SimulationParameters::workSize

the size of OpenCL 1DRange kernel work groups

## 7.29.3.77 qint32 Langmuir::SimulationParameters::workX

the x size of OpenCL 3DRange kernel work groups - only needed if using [SimulationParameters::outputCoulomb](#)

## 7.29.3.78 qint32 Langmuir::SimulationParameters::workY

the y size of OpenCL 3DRange kernel work groups - only needed if using [SimulationParameters::outputCoulomb](#)

## 7.29.3.79 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:

- [parameters.h](#)

## 7.30 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

#### 7.30.1 Detailed Description

A class to inject charges.

#### 7.30.2 Constructor & Destructor Documentation

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

create a [SourceAgent](#)

#### 7.30.3 Member Function Documentation

7.30.3.1 `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](#).

7.30.3.2 `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](#).

7.30.3.3 `int Langmuir::SourceAgent::randomNeighborSiteID ( )` [protected]

choose a random site ID from the neighborlist.

7.30.3.4 `int Langmuir::SourceAgent::randomSiteID ( )` [protected]

choose a random site ID

It can be any possible site in the grid.

7.30.3.5 `bool Langmuir::SourceAgent::shouldTransport ( int site )` [protected],[virtual]

decides if charge should be injected using a constant probability

#### Parameters

|             |                   |
|-------------|-------------------|
| <i>site</i> | 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](#).

7.30.3.6 `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 guaranteed that a charge will be injected.

7.30.3.7 `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.

7.30.3.8 `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.

7.30.3.9 `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 files:

- [sourceagent.h](#)
- [sourceagent.cpp](#)

## 7.31 Langmuir::Tolerance Class Reference

A class to check if the simulation is converging.

```
#include <tolerance.h>
```

### Public Member Functions

- [Tolerance](#) ([World](#) &world, [FluxAgent](#) \*flux, double criteria=0.01, [QObject](#) \*parent=0)  
*Create the tolerance checking object.*
- void [check](#) ()  
*check if the convergence criteria has been met*
- bool [converged](#) ()  
*true if the convergence criteria has been met the appropriate number of times*

### Private Member Functions

- void [checkCriteria](#) ()  
*make sure the criteria is a valid percent*
- void [setPrevious](#) ()  
*set the previous success/step ratio to be whatever flux currently reports*
- void [setCurrent](#) ()  
*set the current success/step ratio to whatever flux currently reports*
- double [percentChange](#) ()  
*calculate the percent change between previous and current flux success/step ratios*

### Private Attributes

- double [m\\_current](#)  
*current flux success/step ratio*
- double [m\\_previous](#)  
*previous flux success/step ratio*
- double [m\\_criteria](#)  
*the maximum percent difference between previous and current for the criteria to be met*
- int [m\\_converged](#)  
*the number of consecutive times the flux agent has met the convergence criteria*
- [FluxAgent](#) \* [m\\_flux](#)  
*the flux agent being tracked*

- [World](#) & [m\\_world](#)

*reference to the world object*

### 7.31.1 Detailed Description

A class to check if the simulation is converging.

### 7.31.2 Constructor & Destructor Documentation

**7.31.2.1** `Langmuir::Tolerance::Tolerance ( World & world, FluxAgent * flux, double criteria = 0.01, QObject * parent = 0 ) [explicit]`

Create the tolerance checking object.

Parameters

|                  |                            |
|------------------|----------------------------|
| <i>reference</i> | to the world object        |
| <i>a</i>         | flux agent to track        |
| <i>the</i>       | criterion for convergence  |
| <i>parent</i>    | the parent of this QObject |

### 7.31.3 Member Function Documentation

**7.31.3.1** `void Langmuir::Tolerance::check ( )`

check if the convergence criteria has been met

**7.31.3.2** `void Langmuir::Tolerance::checkCriteria ( ) [private]`

make sure the criteria is a valid percent

**7.31.3.3** `bool Langmuir::Tolerance::converged ( )`

true if the convergence criteria has been met the appropriate number of times

**7.31.3.4** `double Langmuir::Tolerance::percentChange ( ) [private]`

calculate the percent change between previous and current flux success/step ratios

**7.31.3.5** `void Langmuir::Tolerance::setCurrent ( ) [private]`

set the current success/step ratio to whatever flux currently reports

**7.31.3.6** `void Langmuir::Tolerance::setPrevious ( ) [private]`

set the previous success/step ratio to be whatever flux currently reports

### 7.31.4 Member Data Documentation

#### 7.31.4.1 `int Langmuir::Tolerance::m_converged` [private]

the number of consecutive times the flux agent has met the convergence criteria

#### 7.31.4.2 `double Langmuir::Tolerance::m_criteria` [private]

the maximum percent difference between previous and current for the criteria to be met

#### 7.31.4.3 `double Langmuir::Tolerance::m_current` [private]

current flux success/step ratio

#### 7.31.4.4 `FluxAgent* Langmuir::Tolerance::m_flux` [private]

the flux agent being tracked

#### 7.31.4.5 `double Langmuir::Tolerance::m_previous` [private]

previous flux success/step ratio

#### 7.31.4.6 `World& Langmuir::Tolerance::m_world` [private]

reference to the world object

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

- [tolerance.h](#)
- [tolerance.cpp](#)

## 7.32 `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<>`  
  - virtual 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 keyValuePair () 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

### 7.32.1 Detailed Description

`template<class T>class Langmuir::TypedVariable< T >`

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

## 7.32.2 Constructor & Destructor Documentation

7.32.2.1 `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

|               |  |
|---------------|--|
| <i>key</i>    | the name of the variable   |
| <i>value</i>  | the location where the value of the variable is stored                                   |
| <i>mode</i>   | flags to alter variable's behaviour, see <code>AbstractVariable::VariableModeFlag</code> |
| <i>parent</i> | QObject this belongs to  |

## 7.32.3 Member Function Documentation

7.32.3.1 `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).

7.32.3.2 `template<> void Langmuir::TypedVariable< QString >::convert ( const QString & token, QString & result ) [inline]`

7.32.3.3 `template<> void Langmuir::TypedVariable< qreal >::convert ( const QString & token, qreal & result ) [inline]`

7.32.3.4 `template<> void Langmuir::TypedVariable< float >::convert ( const QString & token, float & result ) [inline]`

7.32.3.5 `template<> void Langmuir::TypedVariable< bool >::convert ( const QString & token, bool & result ) [inline]`

7.32.3.6 `template<> void Langmuir::TypedVariable< qint32 >::convert ( const QString & token, qint32 & result ) [inline]`

7.32.3.7 `template<> void Langmuir::TypedVariable< quint32 >::convert ( const QString & token, quint32 & result ) [inline]`

7.32.3.8 `template<> void Langmuir::TypedVariable< qint64 >::convert ( const QString & token, qint64 & result ) [inline]`

7.32.3.9 `template<> void Langmuir::TypedVariable< quint64 >::convert ( const QString & token, quint64 & result ) [inline]`

7.32.3.10 `template<> void Langmuir::TypedVariable< QDateTime >::convert ( const QString & token, QDateTime & result ) [inline]`

7.32.3.11 `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](#).

7.32.3.12 `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](#).

7.32.3.13 `template<> QString Langmuir::TypedVariable< QString >::keyValue ( ) const [inline],[virtual]`

Get this variable's key and value in the form 'key = value'.

Implements [Langmuir::Variable](#).

7.32.3.14 `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](#).

7.32.3.15 `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](#).

7.32.3.16 `template<> QString Langmuir::TypedVariable< float >::value ( ) const [inline],[virtual]`

Get this variable's value as a QString.

Implements [Langmuir::Variable](#).

7.32.3.17 `template<> QString Langmuir::TypedVariable< qreal >::value ( ) const [inline],[virtual]`

Get this variable's value as a QString.

Implements [Langmuir::Variable](#).

7.32.3.18 `template<> QString Langmuir::TypedVariable< bool >::value ( ) const [inline],[virtual]`

Get this variable's value as a QString.

Implements [Langmuir::Variable](#).

7.32.3.19 `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](#).

## 7.32.4 Member Data Documentation

7.32.4.1 `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:

- [variable.h](#)

## 7.33 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 behavior 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.*

### 7.33.1 Detailed Description

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

### 7.33.2 Member Enumeration Documentation

#### 7.33.2.1 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.

### 7.33.3 Constructor & Destructor Documentation

#### 7.33.3.1 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

### 7.33.4 Member Function Documentation

#### 7.33.4.1 bool Langmuir::Variable::isConstant ( ) const [inline]

True if the [Variable::Constant](#) mode flag was set.

see if the [Variable](#) is really a Constant

#### 7.33.4.2 virtual QString Langmuir::Variable::key ( ) const [pure virtual]

Get this variable's key (name)

Implemented in [Langmuir::TypedVariable< T >](#).

**7.33.4.3** `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 >](#).

**7.33.4.4** `const Variable::VariableMode & Langmuir::Variable::mode ( ) const` `[inline]`

Get this variable's mode flags.

get the mode flags of this [Variable](#)

**7.33.4.5** `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 >](#).

**7.33.4.6** `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 >](#).

**7.33.4.7** `virtual void Langmuir::Variable::write ( QTextStream & stream ) const` `[protected], [pure virtual]`

Write 'key = value' to a stream.

Implemented in [Langmuir::TypedVariable< T >](#).

## 7.33.5 Friends And Related Function Documentation

**7.33.5.1** `QTextStream& operator<< ( QTextStream & stream, const Variable & variable )` `[friend]`

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

**7.33.5.2** `QDebug operator<< ( QDebug dbg, const Variable & variable )` `[friend]`

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

**7.33.5.3** `std::ostream& operator<< ( std::ostream & stream, Variable & variable )` `[friend]`

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

## 7.33.6 Member Data Documentation

**7.33.6.1** `QString Langmuir::Variable::m_key` `[protected]`

The name of this variable.

## 7.33.6.2 VariableMode Langmuir::Variable::m\_mode [protected]

The mode flags for this variable.

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

- [variable.h](#)

## 7.34 Langmuir::World Class Reference

A class to hold all objects in a simulation.

```
#include <world.h>
```

### Public Member Functions

- [World](#) (const QString &fileName, QObject \*parent=0)  
*create a world to simulate in*
- [~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*
- [OpenCIHelper](#) & [opencl](#) ()  
*get the [OpenCIHelper](#), 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 [initialize](#) (const QString &fileName="")  
*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  $\text{erf}(r/(s*\sqrt{2}))$  values*
- [boost::multi\\_array< double, 3 > m\\_sl](#)  
*self interaction, which is  $1/(4 \pi \epsilon_0 r)$ , 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*

### 7.34.1 Detailed Description

A class to hold all objects in a simulation.

### 7.34.2 Constructor & Destructor Documentation

#### 7.34.2.1 `Langmuir::World::World ( const QString & fileName, QObject * parent = 0 )`

create a world to simulate in

##### Parameters

|                 |                         |
|-----------------|-------------------------|
| <i>fileName</i> | the input file name     |
| <i>parent</i>   | QObject this belongs to |

Calls the [initialize\(\)](#) function.

#### 7.34.2.2 `Langmuir::World::~~World ( )`

destroys the entire [World](#), and everything in it...including you.

### 7.34.3 Member Function Documentation

#### 7.34.3.1 `bool Langmuir::World::atMaxCharges ( )`

check if the maximum number of charges has been reached

#### 7.34.3.2 `bool Langmuir::World::atMaxElectrons ( )`

check if the maximum number of electrons has been reached

#### 7.34.3.3 `bool Langmuir::World::atMaxHoles ( )`

check if the maximum number of holes has been reached

#### 7.34.3.4 `bool Langmuir::World::chargesAreBalanced ( )`

true when electrons and holes are balanced

#### 7.34.3.5 `CheckPointer & Langmuir::World::checkPointer ( )`

get the [CheckPointer](#), used for reading and writing input files.

#### 7.34.3.6 `boost::multi_array< double, 3 > & Langmuir::World::couplingConstants ( )`

get the coupling constants

#### 7.34.3.7 `void Langmuir::World::createDrains ( )` `[private]`

create DrainAgents

7.34.3.8 void Langmuir::World::createSources ( ) [private]

create SourceAgents

7.34.3.9 QList< int > & Langmuir::World::defectSiteIDs ( )

get a list of all defect sites

7.34.3.10 QList< DrainAgent \* > & Langmuir::World::drains ( )

get a list of all DrainAgents

7.34.3.11 QList< DrainAgent \* > & Langmuir::World::eDrains ( )

get a list of all ElectronSourceAgents

7.34.3.12 ElectronDrainAgent & Langmuir::World::electronDrainAgentLeft ( )

get the left [ElectronDrainAgent](#)

7.34.3.13 ElectronDrainAgent & Langmuir::World::electronDrainAgentRight ( )

get the right [ElectronDrainAgent](#)

7.34.3.14 Grid & Langmuir::World::electronGrid ( )

get the [Grid](#) used, used for holding ElectronAgents

7.34.3.15 QList< ChargeAgent \* > & Langmuir::World::electrons ( )

get a list of all ElectronAgents

7.34.3.16 int Langmuir::World::electronsMinusHoles ( )

The number of electrons - holes.

7.34.3.17 ElectronSourceAgent & Langmuir::World::electronSourceAgentLeft ( )

get the left [ElectronSourceAgent](#)

7.34.3.18 ElectronSourceAgent & Langmuir::World::electronSourceAgentRight ( )

get the right [ElectronSourceAgent](#)

7.34.3.19 boost::multi\_array< double, 3 > & Langmuir::World::eR ( )

get the array of precomputed  $\text{erf}(r/(\text{sqrt}(2)*\text{sigma}))$

**7.34.3.20** `QList< SourceAgent * > & Langmuir::World::eSources ( )`

get a list of all ElectronSourceAgents

**7.34.3.21** `ExcitonSourceAgent & Langmuir::World::excitonSourceAgent ( )`

get the [RecombinationAgent](#)

**7.34.3.22** `QList< FluxAgent * > & Langmuir::World::fluxes ( )`

get a list of all FluxAgents

**7.34.3.23** `QList< DrainAgent * > & Langmuir::World::hDrains ( )`

get a list of all ElectronSourceAgents

**7.34.3.24** `HoleDrainAgent & Langmuir::World::holeDrainAgentLeft ( )`

get the left [HoleDrainAgent](#)

**7.34.3.25** `HoleDrainAgent & Langmuir::World::holeDrainAgentRight ( )`

get the right [HoleDrainAgent](#)

**7.34.3.26** `Grid & Langmuir::World::holeGrid ( )`

get the hole [Grid](#), used for holding HoleAgents

**7.34.3.27** `QList< ChargeAgent * > & Langmuir::World::holes ( )`

get a list of all HoleAgents

**7.34.3.28** `int Langmuir::World::holesMinusElectrons ( )`

The number of holes - electrons.

**7.34.3.29** `HoleSourceAgent & Langmuir::World::holeSourceAgentLeft ( )`

get the left [HoleSourceAgent](#)

**7.34.3.30** `HoleSourceAgent & Langmuir::World::holeSourceAgentRight ( )`

get the right [HoleSourceAgent](#)

**7.34.3.31** `QList< SourceAgent * > & Langmuir::World::hSources ( )`

get a list of all ElectronSourceAgents

7.34.3.32 `void Langmuir::World::initialize ( const QString & fileName = " " ) [private]`

initialize all objects

#### Parameters

|                 |                 |
|-----------------|-----------------|
| <i>fileName</i> | 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.

7.34.3.33 `boost::multi_array< double, 3 > & Langmuir::World::iR ( )`

get the array of precomputed inverse-r values

7.34.3.34 `KeyValueParser & Langmuir::World::keyValueParser ( )`

get the [KeyValueParser](#), used for parsing input files.

7.34.3.35 `Logger & Langmuir::World::logger ( )`

get the [Logger](#), used for writing output

7.34.3.36 `int Langmuir::World::maxChargeAgents ( )`

get the max number of ChargeAgents allowed

7.34.3.37 `int Langmuir::World::maxChargeAgentsAndChargedDefects ( )`

get the max number of ChargeAgents & charged defects

7.34.3.38 `int Langmuir::World::maxDefects ( )`

get the max number of Defects allowed

7.34.3.39 `int Langmuir::World::maxElectronAgents ( )`

get the max number of ElectronAgents allowed

7.34.3.40 `int Langmuir::World::maxHoleAgents ( )`

get the max number of HoleAgents allowed

7.34.3.41 `int Langmuir::World::maxTraps ( )`

get the max number of Traps allowed

7.34.3.42 `int Langmuir::World::numChargeAgents ( )`

get the current number of ChargeAgents

7.34.3.43 `int Langmuir::World::numChargeAgentsAndChargedDefects ( )`

get the current number of ChargeAgents & charged defects

7.34.3.44 `int Langmuir::World::numDefects ( )`

get the current number of Defects

7.34.3.45 `int Langmuir::World::numElectronAgents ( )`

get the current number of ElectronAgents

7.34.3.46 `int Langmuir::World::numHoleAgents ( )`

get the current number of HoleAgents

7.34.3.47 `int Langmuir::World::numTraps ( )`

get the current number of Traps

7.34.3.48 `OpenCIHelper & Langmuir::World::opencl ( )`

get the [OpenCIHelper](#), used for calculating Coulomb interactions with a Graphics Card

7.34.3.49 `SimulationParameters & Langmuir::World::parameters ( )`

get the [SimulationParameters](#), a struct used for holding simulation parameters.

7.34.3.50 `double Langmuir::World::percentElectronAgents ( )`

get the percent of ElectronAgents reached, of the total grid volume

7.34.3.51 `double Langmuir::World::percentHoleAgents ( )`

get the percent of HoleAgents reached, of the total grid volume

7.34.3.52 `void Langmuir::World::placeDefects ( const QList< int > & siteIDs = QList<int>() ) [private]`

places defects

#### Parameters

|                |                           |
|----------------|---------------------------|
| <i>siteIDs</i> | 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.

7.34.3.53 `void Langmuir::World::placeElectrons ( const QList< int > & siteIDs = QList<int>() ) [private]`

places electrons



## Parameters

|                |                             |
|----------------|-----------------------------|
| <i>siteIDs</i> | a list of electron site ids |
|----------------|-----------------------------|

Places carriers according to the site ids passed. If more need placing (according to [SimulationParameters::seed-Charges](#)), then they are placed randomly.

**7.34.3.54** void Langmuir::World::placeHoles ( const QList< int > & *siteIDs* = QList<int>() ) [private]

places holes

## Parameters

|                |                         |
|----------------|-------------------------|
| <i>siteIDs</i> | a list of hole site ids |
|----------------|-------------------------|

Places carriers according to the site ids passed. If more need placing (according to [SimulationParameters::seed-Charges](#)), then they are placed randomly.

**7.34.3.55** Potential & Langmuir::World::potential ( )

get the [Potential](#), a calculator used for...calculating the potential.

**7.34.3.56** boost::multi\_array< double, 3 > & Langmuir::World::R1 ( )

get the array of precomputed r-squared values

**7.34.3.57** boost::multi\_array< double, 3 > & Langmuir::World::R2 ( )

get the array of precomputed r values

**7.34.3.58** Random & Langmuir::World::randomNumberGenerator ( )

get the [Random](#), used for creating random numbers

**7.34.3.59** double Langmuir::World::reachedChargeAgents ( )

get the percent of ChargeAgents reached, of the max allowed

**7.34.3.60** double Langmuir::World::reachedElectronAgents ( )

get the percent of ElectronAgents reached, of the max allowed

**7.34.3.61** double Langmuir::World::reachedHoleAgents ( )

get the percent of HoleAgents reached, of the max allowed

**7.34.3.62** RecombinationAgent & Langmuir::World::recombinationAgent ( )

get the [RecombinationAgent](#)

**7.34.3.63** `void Langmuir::World::setFluxInfo ( const QList< quint64 > & fluxInfo )` [private]

set attempts / successes for sources / drains

**7.34.3.64** `boost::multi_array< double, 3 > & Langmuir::World::sl ( )`

get the self interactions

**7.34.3.65** `QList< SourceAgent * > & Langmuir::World::sources ( )`

get a list of all SourceAgents

**7.34.3.66** `QList< int > & Langmuir::World::trapSiteIDs ( )`

get a list of all trap sites

**7.34.3.67** `QList< double > & Langmuir::World::trapSitePotentials ( )`

get a list of all trap potentials

**7.34.3.68** `QList< DrainAgent * > & Langmuir::World::xDrains ( )`

get a list of all ElectronSourceAgents

**7.34.3.69** `QList< SourceAgent * > & Langmuir::World::xSources ( )`

get a list of all ElectronSourceAgents

## 7.34.4 Member Data Documentation

**7.34.4.1** `CheckPointer* Langmuir::World::m_checkPointer` [private]

pointer to [CheckPointer](#), used for reading/writing input(checkpoint) files

**7.34.4.2** `boost::multi_array<double,3> Langmuir::World::m_couplingConstants` [private]

array of coupling constants

This array is indexed by dx, dy, dz values.

**7.34.4.3** `QList<int> Langmuir::World::m_defectSiteIDs` [private]

list of defect sites

**7.34.4.4** `QList<DrainAgent*> Langmuir::World::m_drains` [private]

list of DrainAgents

**7.34.4.5** `QList<DrainAgent*> Langmuir::World::m_eDrains` [private]

list of ElectronSourceAgents

**7.34.4.6** `ElectronDrainAgent* Langmuir::World::m_electronDrainAgentLeft` [private]

pointer to left [ElectronDrainAgent](#)

**7.34.4.7** `ElectronDrainAgent* Langmuir::World::m_electronDrainAgentRight` [private]

pointer to right [ElectronDrainAgent](#)

**7.34.4.8** `Grid* Langmuir::World::m_electronGrid` [private]

pointer to electron [Grid](#), used for keeping track of ElectronAgents

**7.34.4.9** `QList<ChargeAgent*> Langmuir::World::m_electrons` [private]

list of electrons

**7.34.4.10** `ElectronSourceAgent* Langmuir::World::m_electronSourceAgentLeft` [private]

pointer to left [ElectronDrainAgent](#)

**7.34.4.11** `ElectronSourceAgent* Langmuir::World::m_electronSourceAgentRight` [private]

pointer to right [ElectronDrainAgent](#)

**7.34.4.12** `boost::multi_array<double,3> Langmuir::World::m_eR` [private]

array of precomputed  $\text{erf}(r/(s*\text{sqrt}(2)))$  values

This array is indexed by dx, dy, dz values, and r is in grid-units

**7.34.4.13** `QList<SourceAgent*> Langmuir::World::m_eSources` [private]

list of ElectronSourceAgents

**7.34.4.14** `ExcitonSourceAgent* Langmuir::World::m_excitonSourceAgent` [private]

pointer to [ExcitonSourceAgent](#), used for injecting Excitons

**7.34.4.15** `QList<FluxAgent*> Langmuir::World::m_fluxAgents` [private]

list of all FluxAgents, such as SoureAgents, DrainAgents, etc.

**7.34.4.16** `QList<DrainAgent*> Langmuir::World::m_hDrains` [private]

list of HoleSourceAgents

**7.34.4.17** `HoleDrainAgent* Langmuir::World::m_holeDrainAgentLeft` `[private]`

pointer to left [HoleDrainAgent](#)

**7.34.4.18** `HoleDrainAgent* Langmuir::World::m_holeDrainAgentRight` `[private]`

pointer to right [HoleDrainAgent](#)

**7.34.4.19** `Grid* Langmuir::World::m_holeGrid` `[private]`

pointer to hole [Grid](#), used for keeping track of HoleAgents

**7.34.4.20** `QList<ChargeAgent*> Langmuir::World::m_holes` `[private]`

list of holes

**7.34.4.21** `HoleSourceAgent* Langmuir::World::m_holeSourceAgentLeft` `[private]`

pointer to left [HoleDrainAgent](#)

**7.34.4.22** `HoleSourceAgent* Langmuir::World::m_holeSourceAgentRight` `[private]`

pointer to right [HoleDrainAgent](#)

**7.34.4.23** `QList<SourceAgent*> Langmuir::World::m_hSources` `[private]`

list of HoleSourceAgents

**7.34.4.24** `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

**7.34.4.25** `KeyValueParser* Langmuir::World::m_keyValueParser` `[private]`

pointer to [KeyValueParser](#), used for parsing key=value pairs

**7.34.4.26** `Logger* Langmuir::World::m_logger` `[private]`

pointer to [Logger](#), used for output

**7.34.4.27** `int Langmuir::World::m_maxDefects` `[private]`

max number of defects

**7.34.4.28** `int Langmuir::World::m_maxElectrons` `[private]`

max number of electrons

7.34.4.29 `int Langmuir::World::m_maxHoles` `[private]`

max number of holes

7.34.4.30 `int Langmuir::World::m_maxTraps` `[private]`

max number of traps

7.34.4.31 `OpenCIHelper* Langmuir::World::m_oci` `[private]`

pointer to [OpenCIHelper](#), used for Graphics Card calculations

7.34.4.32 `SimulationParameters* Langmuir::World::m_parameters` `[private]`

pointer to [SimulationParameters](#)

7.34.4.33 `Potential* Langmuir::World::m_potential` `[private]`

pointer to [Potential](#), used for calculating the potential

7.34.4.34 `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

7.34.4.35 `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

7.34.4.36 `Random* Langmuir::World::m_rand` `[private]`

pointer to [Random](#), used for generating random numbers

7.34.4.37 `RecombinationAgent* Langmuir::World::m_recombinationAgent` `[private]`

pointer to electron/hole [RecombinationAgent](#), used for removing Excitons

7.34.4.38 `boost::multi_array<double,3> Langmuir::World::m_sl` `[private]`

self interaction, which is  $1/(4 \pi e e_0 r)$ , 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.

7.34.4.39 `QList<SourceAgent*> Langmuir::World::m_sources` `[private]`

list of SourceAgents

7.34.4.40 `QList<int> Langmuir::World::m_trapSiteIDs` [private]

list of trap sites

7.34.4.41 `QList<double> Langmuir::World::m_trapSitePotentials` [private]

list of trap potentials

7.34.4.42 `QList<DrainAgent*> Langmuir::World::m_xDrains` [private]

list of ExcitonSourceAgents

7.34.4.43 `QList<SourceAgent*> Langmuir::World::m_xSources` [private]

list of ExcitonSourceAgents

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

- [world.h](#)
- [world.cpp](#)

## 7.35 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*

#### 7.35.1 Detailed Description

A class to output xyz files.

## 7.35.2 Constructor & Destructor Documentation

### 7.35.2.1 Langmuir::XYZWriter::XYZWriter ( World & world, const QString & name, QObject \* parent = 0 )

constructs the writer, has the same parameters as [OutputInfo](#)

## 7.35.3 Member Function Documentation

### 7.35.3.1 void Langmuir::XYZWriter::write ( )

Write XYZ of the current step to the stream.

### 7.35.3.2 void Langmuir::XYZWriter::writeVMDInitFile ( ) [protected]

write a VMD script useful for opening the XYZ file

## 7.35.4 Member Data Documentation

### 7.35.4.1 OutputStream Langmuir::XYZWriter::m\_stream [protected]

output file stream

### 7.35.4.2 World& Langmuir::XYZWriter::m\_world [protected]

reference to the world object

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

- [writer.h](#)
- [writer.cpp](#)





## Chapter 8

# File Documentation

### 8.1 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

- namespace [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.*

### 8.2 chargeagent.cpp File Reference

```
#include "openclhelper.h"
```

```
#include "chargeagent.h"
#include "drainagent.h"
#include "parameters.h"
#include "simulation.h"
#include "potential.h"
#include "cubicgrid.h"
#include "world.h"
#include "rand.h"
```

## Namespaces

- namespace [Langmuir](#)

## 8.3 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

- namespace [Langmuir](#)

## 8.4 checkpointer.cpp File Reference

```
#include "checkpointer.h"
#include "chargeagent.h"
#include "output.h"
#include "world.h"
#include "rand.h"
#include "keyvalueparser.h"
#include "fluxagent.h"
#include <fstream>
#include <limits>
#include <iomanip>
```

## Namespaces

- namespace [Langmuir](#)

## 8.5 checkpointer.h File Reference

```
#include <QObject>
#include <QMap>
#include "parameters.h"
```

### Classes

- class [Langmuir::CheckPointer](#)  
*A class to read and write checkpoint files.*

### Namespaces

- namespace [Langmuir](#)

### Functions

- static std::ostream & [Langmuir::operator<<](#) (std::ostream &stream, QString &string)
- static std::istream & [Langmuir::operator>>](#) (std::istream &stream, QString &string)

## 8.6 copy\_to\_frank.py File Reference

### Namespaces

- namespace [copy\\_to\\_frank](#)

### Variables

- tuple [copy\\_to\\_frank.work](#) = os.getcwd()
- list [copy\\_to\\_frank.exclude](#)
- list [copy\\_to\\_frank.found](#) = []
- [copy\\_to\\_frank.copy\\_me](#) = True

## 8.7 cubicgrid.cpp File Reference

```
#include "cubicgrid.h"
#include <cmath>
#include "world.h"
#include "parameters.h"
#include "drainagent.h"
```

### Namespaces

- namespace [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.*

## 8.8 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

- namespace [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.*

## 8.9 drainagent.cpp File Reference

```
#include "drainagent.h"
#include "chargeagent.h"
#include "parameters.h"
#include "writer.h"
#include "world.h"
#include "rand.h"
```

## Namespaces

- namespace [Langmuir](#)

## 8.10 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

- namespace [Langmuir](#)

## 8.11 fluxagent.cpp File Reference

```
#include "fluxagent.h"  
#include "parameters.h"  
#include "world.h"  
#include "rand.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.12 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

- namespace [Langmuir](#)

## 8.13 gridview.cpp File Reference

```
#include "openclhelper.h"  
#include "drainagent.h"  
#include "gridview.h"  
#include "GL/glu.h"  
#include "writer.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.14 keyvalueparser.cpp File Reference

```
#include "keyvalueparser.h"  
#include <QDebug>  
#include <QRegExp>  
#include <QStringList>  
#include <QFile>  
#include "output.h"  
#include <ostream>
```

### Namespaces

- namespace [Langmuir](#)

### Functions

- `std::ostream & Langmuir::operator<< (std::ostream &stream, const KeyValueCollection &keyValueCollection)`

## 8.15 keyvalueparser.h File Reference

```
#include <QObject>  
#include "variable.h"  
#include "parameters.h"  
#include "world.h"
```

### Classes

- class [Langmuir::KeyValueCollection](#)  
*A class to read the parameters and store them in the correct place.*

### Namespaces

- namespace [Langmuir](#)

## 8.16 langmuir.cpp File Reference

```
#include "sourceagent.h"
#include "simulation.h"
#include "drainagent.h"
#include "cubicgrid.h"
#include "keyvalueparser.h"
#include "writer.h"
#include "world.h"
#include "openclhelper.h"
#include "checkpointinter.h"
#include "parameters.h"
#include <QApplication>
#include <QPrinter>
#include <QPainter>
#include <QColor>
#include <QtCore/QStringList>
#include <QtCore/QFile>
#include <QtCore/QTextStream>
#include <QtCore/QDateTime>
#include <QtCore/QDebug>
#include <QThreadPool>
```

### Functions

- `QTextStream & progress` (`QTextStream &stream`, `SimulationParameters &par`)
- `void alterMaxThreads` (`SimulationParameters &par`)
- `int main` (`int argc`, `char *argv[]`)

#### 8.16.1 Function Documentation

8.16.1.1 `void alterMaxThreads ( SimulationParameters & par )`

8.16.1.2 `int main ( int argc, char * argv[] )`

8.16.1.3 `QTextStream& progress ( QTextStream & stream, SimulationParameters & par )`

## 8.17 langmuirview.cpp File Reference

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

### Functions

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

#### 8.17.1 Function Documentation

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

## 8.18 opcnlhelper.cpp File Reference

```
#include <QTextStream>
#include "pbsgpuparser.h"
#include "opcnlhelper.h"
#include "chargeagent.h"
#include "parameters.h"
#include "cubicgrid.h"
#include "potential.h"
#include "world.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.19 opcnlhelper.h File Reference

```
#include <QObject>
#include <QVector>
```

### Classes

- class [Langmuir::OpenCLHelper](#)  
*A Class to run OpenCL calculations.*

### Namespaces

- namespace [Langmuir](#)

### Macros

- `#define` [\\_\\_CL\\_ENABLE\\_EXCEPTIONS](#)

### 8.19.1 Macro Definition Documentation

#### 8.19.1.1 `#define` [\\_\\_CL\\_ENABLE\\_EXCEPTIONS](#)

## 8.20 output.cpp File Reference

```
#include "output.h"
#include <QDateTime>
#include <QFileInfo>
#include <QDir>
```

### Namespaces

- namespace [Langmuir](#)



## Functions

- void [Langmuir::backupFile](#) (const QString &name)  
*Back up a file.*
- 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*

### 8.20.1 Function Documentation

#### 8.20.1.1 QTextStream& newline ( QTextStream & s )

put a newline character in the stream that ignores the streams current FieldWidth

#### 8.20.1.2 QTextStream& space ( QTextStream & s )

put a space in the stream that ignores the streams current FieldWidth

## 8.21 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

- namespace [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.*

### 8.21.1 Function Documentation

#### 8.21.1.1 QTextStream& newline ( QTextStream & s )

put a newline character in the stream that ignores the streams current FieldWidth

#### 8.21.1.2 QTextStream& space ( QTextStream & s )

put a space in the stream that ignores the streams current FieldWidth

## 8.22 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

- namespace [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*

## 8.23 pbsgpuparser.cpp File Reference

```
#include "pbsgpuparser.h"
#include <QRegExp>
#include <QDebug>
#include <QFile>
```

### Namespaces

- namespace [Langmuir](#)

## 8.24 pbsgpuparser.h File Reference

```
#include <QObject>
#include <QList>
```

### Classes

- class [Langmuir::PBSGPUParser](#)

### Namespaces

- namespace [Langmuir](#)

## 8.25 potential.cpp File Reference

```
#include "potential.h"
#include "parameters.h"
#include "chargeagent.h"
#include "cubicgrid.h"
#include "world.h"
#include "rand.h"
#include <cmath>
```

### Namespaces

- namespace [Langmuir](#)

## 8.26 potential.h File Reference

```
#include <QObject>
#include "boost/multi_array.hpp"
```

### Classes

- class [Langmuir::Potential](#)  
*A class to calculate the potential.*

### Namespaces

- namespace [Langmuir](#)

### Macros

- `#define` [BOOST\\_DISABLE\\_ASSERTS](#)

## 8.26.1 Macro Definition Documentation

### 8.26.1.1 `#define BOOST_DISABLE_ASSERTS`

## 8.27 rand.cpp File Reference

```
#include "rand.h"
#include <fstream>
#include <sstream>
```

### Namespaces

- namespace [Langmuir](#)

### Functions

- `QDataStream & Langmuir::operator<<` (`QDataStream &stream`, `Random &random`)
- `QDataStream & Langmuir::operator>>` (`QDataStream &stream`, `Random &random`)
- `QTextStream & Langmuir::operator<<` (`QTextStream &stream`, `Random &random`)
- `QTextStream & Langmuir::operator>>` (`QTextStream &stream`, `Random &random`)
- `std::ostream & Langmuir::operator<<` (`std::ostream &stream`, `Random &random`)
- `std::istream & Langmuir::operator>>` (`std::istream &stream`, `Random &random`)

## 8.28 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

- namespace [Langmuir](#)

## 8.29 README.md File Reference

## 8.30 simulation.cpp File Reference

```
#include "simulation.h"
#include "openclhelper.h"
#include "parameters.h"
#include "chargeagent.h"
#include "sourceagent.h"
#include "drainagent.h"
#include "potential.h"
#include "cubicgrid.h"
#include "checkpointinter.h"
#include "writer.h"
#include "world.h"
#include "rand.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.31 simulation.h File Reference

```
#include <QObject>
```

### Classes

- class [Langmuir::Simulation](#)  
*A class to orchestrate the calculation.*

### Namespaces

- namespace [Langmuir](#)

## 8.32 sourceagent.cpp File Reference

```
#include "sourceagent.h"
#include "chargeagent.h"
#include "parameters.h"
#include "potential.h"
#include "world.h"
#include "rand.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.33 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

- namespace [Langmuir](#)

## 8.34 test.cpp File Reference

```
#include <QtCore>  
#include <iostream>  
#include <fstream>  
#include <boost/random.hpp>  
#include <ctime>
```

### Functions

- int [main](#) (int argc, char \*argv[])

### 8.34.1 Function Documentation

8.34.1.1 int main ( int *argc*, char \* *argv*[] )

## 8.35 tolerance.cpp File Reference

```
#include "tolerance.h"  
#include "fluxagent.h"  
#include "world.h"  
#include "parameters.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.36 tolerance.h File Reference

```
#include <QObject>
```

### Classes

- class [Langmuir::Tolerance](#)  
*A class to check if the simulation is converging.*

### Namespaces

- namespace [Langmuir](#)

## 8.37 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

- namespace [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.*

## 8.38 world.cpp File Reference

```
#include "parameters.h"
#include "openclhelper.h"
#include "chargeagent.h"
#include "sourceagent.h"
#include "drainagent.h"
#include "potential.h"
#include "cubicgrid.h"
#include "writer.h"
#include "world.h"
#include "rand.h"
#include "fluxagent.h"
#include "output.h"
#include "keyvalueparser.h"
#include "checkpointinter.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.39 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

- namespace [Langmuir](#)

### Macros

- `#define` [BOOST\\_DISABLE\\_ASSERTS](#)

### 8.39.1 Macro Definition Documentation

#### 8.39.1.1 `#define` BOOST\_DISABLE\_ASSERTS



## 8.40 writer.cpp File Reference

```
#include "writer.h"  
#include "parameters.h"  
#include "world.h"  
#include "cubicgrid.h"  
#include "chargeagent.h"  
#include "fluxagent.h"  
#include "openclhelper.h"
```

### Namespaces

- namespace [Langmuir](#)

## 8.41 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

- namespace [Langmuir](#)

# Index

- ~Agent
  - Langmuir::Agent, [19](#)
- ~ChargeAgent
  - Langmuir::ChargeAgent, [23](#)
- ~FluxAgent
  - Langmuir::FluxAgent, [42](#)
- ~Grid
  - Langmuir::Grid, [51](#)
- ~KeyValueParser
  - Langmuir::KeyValueParser, [66](#)
- ~OutputStream
  - Langmuir::OutputStream, [75](#)
- ~Random
  - Langmuir::Random, [83](#)
- ~Simulation
  - Langmuir::Simulation, [88](#)
- ~World
  - Langmuir::World, [116](#)
- addToPotential
  - Langmuir::Grid, [51](#)
- Agent
  - Langmuir::Agent, [18](#)
- agent.h, [129](#)
- agentAddress
  - Langmuir::Grid, [51](#)
- agentType
  - Langmuir::Grid, [51](#)
- alterMaxThreads
  - langmuir.cpp, [135](#)
- acquirePath
  - Langmuir::PBSGPUParser, [77](#)
- atMaxCharges
  - Langmuir::World, [116](#)
- atMaxElectrons
  - Langmuir::World, [116](#)
- atMaxHoles
  - Langmuir::World, [116](#)
- attempts
  - Langmuir::FluxAgent, [42](#)
- attemptsSinceLast
  - Langmuir::FluxAgent, [42](#)
- Back
  - Langmuir::Grid, [50](#)
- backupFile
  - Langmuir, [14](#)
- balanceCharges
  - Langmuir::Simulation, [88](#)
  - Langmuir::SimulationParameters, [93](#)
- bindingPotential
  - Langmuir::ChargeAgent, [23](#)
  - Langmuir::ElectronAgent, [34](#)
  - Langmuir::HoleAgent, [62](#)
- boltzmannConstant
  - Langmuir::SimulationParameters, [93](#)
- Bottom
  - Langmuir::Grid, [50](#)
- CarrierWriter
  - Langmuir::CarrierWriter, [21](#)
- charge
  - Langmuir::ChargeAgent, [23](#)
- ChargeAgent
  - Langmuir::ChargeAgent, [23](#)
- chargeAgentCoulombInteractionQtConcurrentCPU
  - Langmuir::Simulation, [88](#)
- chargeAgentCoulombInteractionQtConcurrentGPU
  - Langmuir::Simulation, [88](#)
- chargeagent.cpp, [129](#)
- chargeagent.h, [130](#)
- chargesAreBalanced
  - Langmuir::World, [116](#)
- check
  - Langmuir::Tolerance, [103](#)
- checkCriteria
  - Langmuir::Tolerance, [103](#)
- CheckPointer
  - Langmuir::CheckPointer, [28](#)
- checkPointer
  - Langmuir::World, [116](#)
- checkSimulationParameters
  - Langmuir, [14](#)
- checkStream
  - Langmuir::CheckPointer, [28](#)
- checkpointer.cpp, [130](#)
- checkpointer.h, [131](#)
- chooseFuture
  - Langmuir::ChargeAgent, [23](#)
- chooseNo
  - Langmuir::Random, [83](#)
- chooseSite
  - Langmuir::ExcitonSourceAgent, [38](#)
  - Langmuir::SourceAgent, [100](#)
- chooseYes
  - Langmuir::Random, [83](#)
- compareCoulomb
  - Langmuir::ChargeAgent, [23](#)
- compareHostAndDeviceForAllCarriers
  - Langmuir::OpenCIHelper, [72](#)

- completeTick
  - Langmuir::ChargeAgent, 23
- Constant
  - Langmuir::Variable, 109
- converged
  - Langmuir::Tolerance, 103
- convert
  - Langmuir::TypedVariable, 106
- copy\_me
  - copy\_to\_frank, 11
- copy\_to\_frank, 11
  - copy\_me, 11
  - exclude, 11
  - found, 11
  - work, 11
- copy\_to\_frank.py, 131
- copySiteAndChargeToHostVector
  - Langmuir::OpenCIHelper, 72
- coulombCPU
  - Langmuir::ChargeAgent, 23
- coulombCarriers
  - Langmuir::SimulationParameters, 93
- coulombD
  - Langmuir::Potential, 79
- coulombE
  - Langmuir::Potential, 79
- coulombGPU
  - Langmuir::ChargeAgent, 24
- coulombGaussianSigma
  - Langmuir::SimulationParameters, 93
- coulombH
  - Langmuir::Potential, 79
- coulombImageD
  - Langmuir::Potential, 79
- coulombImageE
  - Langmuir::Potential, 80
- coulombImageH
  - Langmuir::Potential, 80
- coulombInteraction
  - Langmuir::ChargeAgent, 24
- couplingConstants
  - Langmuir::World, 116
- createDrains
  - Langmuir::World, 116
- createSources
  - Langmuir::World, 116
- CubeFace
  - Langmuir::Grid, 50
- cubicgrid.cpp, 131
- cubicgrid.h, 132
- currentStep
  - Langmuir::SimulationParameters, 93
- decideFuture
  - Langmuir::ChargeAgent, 24
- Defect
  - Langmuir::Agent, 18
- defectPercentage
  - Langmuir::SimulationParameters, 93
- defectSiteIDs
  - Langmuir::World, 117
- Defects
  - Langmuir::CheckPointer, 28
- defects
  - Langmuir::ConfigurationInfo, 32
- defectsCharge
  - Langmuir::SimulationParameters, 93
- dielectricConstant
  - Langmuir::SimulationParameters, 93
- Drain
  - Langmuir::Agent, 18
- DrainAgent
  - Langmuir::DrainAgent, 33
- drainRate
  - Langmuir::SimulationParameters, 93
- drainagent.cpp, 132
- drainagent.h, 133
- drains
  - Langmuir::World, 117
- drawCharges
  - Langmuir::GridImage, 61
- drawSites
  - Langmuir::GridImage, 61
- eDrainLRate
  - Langmuir::SimulationParameters, 93
- eDrainRRate
  - Langmuir::SimulationParameters, 93
- eDrains
  - Langmuir::World, 117
- eR
  - Langmuir::World, 117
- eSourceLRate
  - Langmuir::SimulationParameters, 94
- eSourceRRate
  - Langmuir::SimulationParameters, 94
- eSources
  - Langmuir::World, 117
- Electron
  - Langmuir::Agent, 18
- ElectronAgent
  - Langmuir::ElectronAgent, 34
- ElectronDrainAgent
  - Langmuir::ElectronDrainAgent, 35
- electronDrainAgentLeft
  - Langmuir::World, 117
- electronDrainAgentRight
  - Langmuir::World, 117
- electronGrid
  - Langmuir::World, 117
- electronPercentage
  - Langmuir::SimulationParameters, 93
- ElectronSourceAgent
  - Langmuir::ElectronSourceAgent, 36
- electronSourceAgentLeft
  - Langmuir::World, 117
- electronSourceAgentRight
  - Langmuir::World, 117

- Electrons
  - Langmuir::CheckPointer, [28](#)
- electrons
  - Langmuir::ConfigurationInfo, [32](#)
  - Langmuir::World, [117](#)
- electronsMinusHoles
  - Langmuir::World, [117](#)
- electrostaticCutoff
  - Langmuir::SimulationParameters, [94](#)
- electrostaticPrefactor
  - Langmuir::SimulationParameters, [94](#)
- elementaryCharge
  - Langmuir::SimulationParameters, [94](#)
- Empty
  - Langmuir::Agent, [18](#)
- energyChange
  - Langmuir::ElectronDrainAgent, [36](#)
  - Langmuir::ElectronSourceAgent, [37](#)
  - Langmuir::ExcitonSourceAgent, [38](#)
  - Langmuir::FluxAgent, [42](#)
  - Langmuir::HoleDrainAgent, [64](#)
  - Langmuir::HoleSourceAgent, [65](#)
  - Langmuir::RecombinationAgent, [86](#)
- excitonBinding
  - Langmuir::SimulationParameters, [94](#)
- ExcitonSourceAgent
  - Langmuir::ExcitonSourceAgent, [38](#)
- excitonSourceAgent
  - Langmuir::World, [118](#)
- ExcitonWriter
  - Langmuir::ExcitonWriter, [39](#)
- exclude
  - copy\_to\_frank, [11](#)
- face
  - Langmuir::FluxAgent, [42](#)
- faceToLetter
  - Langmuir::FluxAgent, [42](#)
- file
  - Langmuir::OutputStream, [75](#)
- FluxState
  - Langmuir::CheckPointer, [28](#)
- FluxAgent
  - Langmuir::FluxAgent, [41](#)
- fluxInfo
  - Langmuir::ConfigurationInfo, [32](#)
- FluxWriter
  - Langmuir::FluxWriter, [46](#)
- fluxagent.cpp, [133](#)
- fluxagent.h, [133](#)
- fluxes
  - Langmuir::World, [118](#)
- found
  - copy\_to\_frank, [11](#)
- Front
  - Langmuir::Grid, [50](#)
- gaussD
  - Langmuir::Potential, [80](#)
- gaussE
  - Langmuir::Potential, [80](#)
- gaussH
  - Langmuir::Potential, [80](#)
- gaussImageD
  - Langmuir::Potential, [80](#)
- gaussImageE
  - Langmuir::Potential, [81](#)
- gaussImageH
  - Langmuir::Potential, [81](#)
- gaussianStdev
  - Langmuir::SimulationParameters, [94](#)
- generationRate
  - Langmuir::SimulationParameters, [94](#)
- generator01
  - Langmuir::Random, [85](#)
- getCurrentSite
  - Langmuir::Agent, [19](#)
- getFutureSite
  - Langmuir::Agent, [19](#)
- getGrid
  - Langmuir::ChargeAgent, [24](#)
- getIndexS
  - Langmuir::Grid, [51](#)
- getIndexX
  - Langmuir::Grid, [52](#)
- getIndexY
  - Langmuir::Grid, [52](#)
- getIndexZ
  - Langmuir::Grid, [52](#)
- getNeighbors
  - Langmuir::Agent, [19](#)
- getOpenCLID
  - Langmuir::ChargeAgent, [24](#)
- getOutputHost
  - Langmuir::OpenCIHelper, [72](#)
- getOutputHostFuture
  - Langmuir::OpenCIHelper, [72](#)
- getPositionX
  - Langmuir::Grid, [52](#)
- getPositionY
  - Langmuir::Grid, [53](#)
- getPositionZ
  - Langmuir::Grid, [53](#)
- getSpecialAgentList
  - Langmuir::Grid, [53](#)
- getType
  - Langmuir::Agent, [19](#)
- getVariable
  - Langmuir::KeyValueParser, [67](#)
- getWorld
  - Langmuir::Agent, [19](#)
- gpu
  - Langmuir::PBSGPUParser, [77](#)
- gpus
  - Langmuir::PBSGPUParser, [77](#)
- Grid
  - Langmuir::Grid, [50](#)

- grid
  - Langmuir::FluxAgent, 42
- gridFactor
  - Langmuir::SimulationParameters, 94
- GridImage
  - Langmuir::GridImage, 60
- gridX
  - Langmuir::SimulationParameters, 94
- gridY
  - Langmuir::SimulationParameters, 94
- gridZ
  - Langmuir::SimulationParameters, 94
- gridview.cpp, 134
- guessProbability
  - Langmuir::RecombinationAgent, 86
- hDrainLRate
  - Langmuir::SimulationParameters, 95
- hDrainRRate
  - Langmuir::SimulationParameters, 95
- hDrains
  - Langmuir::World, 118
- hSourceLRate
  - Langmuir::SimulationParameters, 95
- hSourceRRate
  - Langmuir::SimulationParameters, 95
- hSources
  - Langmuir::World, 118
- Hole
  - Langmuir::Agent, 18
- HoleAgent
  - Langmuir::HoleAgent, 62
- HoleDrainAgent
  - Langmuir::HoleDrainAgent, 63
- holeDrainAgentLeft
  - Langmuir::World, 118
- holeDrainAgentRight
  - Langmuir::World, 118
- holeGrid
  - Langmuir::World, 118
- holePercentage
  - Langmuir::SimulationParameters, 95
- HoleSourceAgent
  - Langmuir::HoleSourceAgent, 64
- holeSourceAgentLeft
  - Langmuir::World, 118
- holeSourceAgentRight
  - Langmuir::World, 118
- Holes
  - Langmuir::CheckPointer, 28
- holes
  - Langmuir::ConfigurationInfo, 32
  - Langmuir::World, 118
- holesMinusElectrons
  - Langmuir::World, 118
- hoppingRange
  - Langmuir::SimulationParameters, 95
- iR
  - Langmuir::World, 119
- imageCarriers
  - Langmuir::SimulationParameters, 95
- imageDefects
  - Langmuir::SimulationParameters, 95
- imageTraps
  - Langmuir::SimulationParameters, 95
- info
  - Langmuir::OutputStream, 75
- initialize
  - Langmuir::Logger, 69
  - Langmuir::World, 118
- initializeOpenCL
  - Langmuir::OpenCLHelper, 72
- initializeSite
  - Langmuir::FluxAgent, 42
- inject
  - Langmuir::ElectronSourceAgent, 37
  - Langmuir::ExcitonSourceAgent, 38
  - Langmuir::HoleSourceAgent, 65
  - Langmuir::SourceAgent, 100
- integer
  - Langmuir::Random, 83
- inverseKT
  - Langmuir::SimulationParameters, 95
- isConstant
  - Langmuir::Variable, 109
- iterationsPrint
  - Langmuir::SimulationParameters, 95
- iterationsReal
  - Langmuir::SimulationParameters, 95
- key
  - Langmuir::TypedVariable, 106
  - Langmuir::Variable, 109
- keyValue
  - Langmuir::TypedVariable, 107
  - Langmuir::Variable, 109
- KeyValueParser
  - Langmuir::KeyValueParser, 66
- keyValueParser
  - Langmuir::World, 119
- keyvalueparser.cpp, 134
- keyvalueparser.h, 134
- Langmuir, 11
  - backupFile, 14
  - checkSimulationParameters, 14
  - operator<<, 14, 15
  - operator>>, 15
  - setCalculatedValues, 15
- langmuir.cpp, 135
  - alterMaxThreads, 135
  - main, 135
  - progress, 135
- Langmuir::Agent
  - Defect, 18
  - Drain, 18
  - Electron, 18

- Empty, 18
- Hole, 18
- SIZE, 18
- Source, 18
- Langmuir::CheckPointer
  - Defects, 28
  - Electrons, 28
  - FluxState, 28
  - Holes, 28
  - Parameters, 28
  - RandomState, 28
  - TrapPotentials, 28
  - Traps, 28
- Langmuir::Grid
  - Back, 50
  - Bottom, 50
  - Front, 50
  - Left, 50
  - NoFace, 50
  - Right, 50
  - Top, 50
- Langmuir::Variable
  - Constant, 109
- Langmuir::Agent, 17
  - ~Agent, 19
  - Agent, 18
  - getCurrentSite, 19
  - getFutureSite, 19
  - getNeighbors, 19
  - getType, 19
  - getWorld, 19
  - m\_fSite, 20
  - m\_neighbors, 20
  - m\_site, 20
  - m\_type, 20
  - m\_world, 20
  - setCurrentSite, 19
  - setFutureSite, 19
  - setNeighbors, 19
  - toQString, 19
  - Type, 18
- Langmuir::CarrierWriter, 20
  - CarrierWriter, 21
  - m\_stream, 21
  - m\_world, 21
  - write, 21
- Langmuir::ChargeAgent, 21
  - ~ChargeAgent, 23
  - bindingPotential, 23
  - charge, 23
  - ChargeAgent, 23
  - chooseFuture, 23
  - compareCoulomb, 23
  - completeTick, 23
  - coulombCPU, 23
  - coulombGPU, 24
  - coulombInteraction, 24
  - decideFuture, 24
  - getGrid, 24
  - getOpenCLID, 24
  - lifetime, 24
  - m\_charge, 25
  - m\_de, 25
  - m\_grid, 25
  - m\_lifetime, 26
  - m\_openCLID, 26
  - m\_pathlength, 26
  - m\_removed, 26
  - otherGrid, 24
  - otherType, 25
  - pathlength, 25
  - removed, 25
  - setOpenCLID, 25
  - setRemoved, 25
- Langmuir::CheckPointer, 26
  - CheckPointer, 28
  - checkStream, 28
  - load, 28
  - loadDefects, 28
  - loadElectrons, 29
  - loadFluxState, 29
  - loadHoles, 29
  - loadParameters, 29
  - loadRandomState, 29
  - loadTrapPotentials, 29
  - loadTraps, 30
  - m\_world, 31
  - save, 30
  - saveDefects, 30
  - saveElectrons, 30
  - saveFluxState, 30
  - saveHoles, 30
  - saveParameters, 31
  - saveRandomState, 31
  - saveTrapPotentials, 31
  - saveTraps, 31
  - Section, 28
- Langmuir::ConfigurationInfo, 32
  - defects, 32
  - electrons, 32
  - fluxInfo, 32
  - holes, 32
  - trapPotentials, 32
  - traps, 32
- Langmuir::DrainAgent, 33
  - DrainAgent, 33
  - tryToAccept, 33
- Langmuir::ElectronAgent, 33
  - bindingPotential, 34
  - ElectronAgent, 34
  - otherGrid, 34
  - otherType, 34
- Langmuir::ElectronDrainAgent, 35
  - ElectronDrainAgent, 35
  - energyChange, 36
- Langmuir::ElectronSourceAgent, 36

- ElectronSourceAgent, 36
- energyChange, 37
- inject, 37
- validToInject, 37
- Langmuir::ExcitonSourceAgent, 37
  - chooseSite, 38
  - energyChange, 38
  - ExcitonSourceAgent, 38
  - inject, 38
  - shouldTransport, 38
  - validToInject, 38
- Langmuir::ExcitonWriter, 38
  - ExcitonWriter, 39
  - m\_stream, 39
  - m\_world, 39
  - write, 39
- Langmuir::FluxAgent, 39
  - ~FluxAgent, 42
  - attempts, 42
  - attemptsSinceLast, 42
  - energyChange, 42
  - face, 42
  - faceToLetter, 42
  - FluxAgent, 41
  - grid, 42
  - initializeSite, 42
  - m\_attempts, 45
  - m\_face, 45
  - m\_grid, 45
  - m\_lastAttempts, 45
  - m\_lastStep, 45
  - m\_lastSuccesses, 45
  - m\_potential, 46
  - m\_probability, 46
  - m\_successes, 46
  - potential, 43
  - rate, 43
  - resetCounters, 43
  - setAttempts, 43
  - setPotential, 43
  - setRate, 43
  - setRateSmartly, 43
  - setSuccesses, 44
  - shouldTransport, 44
  - stepsSinceLast, 44
  - storeLast, 44
  - successProbability, 45
  - successProbabilitySinceLast, 45
  - successRate, 45
  - successRateSinceLast, 45
  - successes, 44
  - successesSinceLast, 44
- Langmuir::FluxWriter, 46
  - FluxWriter, 46
  - m\_stream, 47
  - m\_world, 47
  - write, 47
- Langmuir::Grid, 47
  - ~Grid, 51
  - addToPotential, 51
  - agentAddress, 51
  - agentType, 51
  - CubeFace, 50
  - getIndexS, 51
  - getIndexX, 52
  - getIndexY, 52
  - getIndexZ, 52
  - getPositionX, 52
  - getPositionY, 53
  - getPositionZ, 53
  - getSpecialAgentList, 53
  - Grid, 50
  - m\_agentType, 58
  - m\_agents, 58
  - m\_potentials, 59
  - m\_specialAgentCount, 59
  - m\_specialAgentReserve, 59
  - m\_specialAgents, 59
  - m\_volume, 59
  - m\_world, 59
  - m\_xSize, 59
  - m\_xyPlaneArea, 59
  - m\_xzPlaneArea, 59
  - m\_ySize, 59
  - m\_yzPlaneArea, 59
  - m\_zSize, 60
  - neighborsFace, 53
  - neighborsSite, 53
  - potential, 53
  - registerAgent, 54
  - registerDefect, 54
  - registerSpecialAgent, 54
  - setPotential, 54
  - sliceIndex, 54
  - specialAgentCount, 55
  - toQString, 55
  - totalDistance, 55
  - unregisterAgent, 55
  - unregisterDefect, 55
  - unregisterSpecialAgent, 55
  - volume, 55
  - xDistance, 55
  - xDistancei, 56
  - xImageDistance, 56
  - xImageDistancei, 56
  - xSize, 56
  - xyPlaneArea, 56
  - yDistance, 56
  - yDistancei, 57
  - yImageDistance, 57
  - yImageDistancei, 57
  - ySize, 57
  - zDistance, 57
  - zDistancei, 57
  - zImageDistance, 58
  - zImageDistancei, 58

- zSize, 58
- Langmuir::GridImage, 60
  - drawCharges, 61
  - drawSites, 61
  - GridImage, 60
  - m\_image, 61
  - mPainter, 61
  - m\_world, 61
  - save, 61
- Langmuir::HoleAgent, 62
  - bindingPotential, 62
  - HoleAgent, 62
  - otherGrid, 62
  - otherType, 63
- Langmuir::HoleDrainAgent, 63
  - energyChange, 64
  - HoleDrainAgent, 63
- Langmuir::HoleSourceAgent, 64
  - energyChange, 65
  - HoleSourceAgent, 64
  - inject, 65
  - validToInject, 65
- Langmuir::KeyValueParser, 65
  - ~KeyValueParser, 66
  - getVariable, 67
  - KeyValueParser, 66
  - m\_orderedNames, 67
  - m\_parameters, 67
  - m\_variableMap, 67
  - m\_world, 67
  - operator<<, 67
  - parameters, 67
  - parse, 67
  - registerVariable, 67
  - save, 67
- Langmuir::Logger, 68
  - initialize, 69
  - Logger, 69
  - m\_carrierWriter, 70
  - m\_excitonWriter, 70
  - m\_fluxWriter, 70
  - m\_world, 70
  - m\_xyzWriter, 71
  - reportCarrier, 69
  - reportExciton, 69
  - reportFluxStream, 69
  - reportXYZStream, 69
  - saveCarriersImage, 69
  - saveCoulombEnergy, 70
  - saveDefectImage, 70
  - saveElectronImage, 70
  - saveGridPotential, 70
  - saveHoleImage, 70
  - saveImage, 70
  - saveTrapImage, 70
- Langmuir::OpenCIHelper, 71
  - compareHostAndDeviceForAllCarriers, 72
  - copySiteAndChargeToHostVector, 72
  - getOutputHost, 72
  - getOutputHostFuture, 72
  - initializeOpenCL, 72
  - launchCoulombKernel1, 73
  - launchCoulombKernel2, 73
  - launchGaussKernel1, 73
  - launchGaussKernel2, 73
  - m\_world, 73
  - OpenCIHelper, 72
  - toggleOpenCL, 73
- Langmuir::OutputInfo, 73
  - OutputInfo, 74
- Langmuir::OutputStream, 74
  - ~OutputStream, 75
  - file, 75
  - info, 75
  - m\_file, 75
  - m\_info, 75
  - OutputStream, 75
- Langmuir::PBSGPUParser, 76
  - acquirePath, 77
  - gpu, 77
  - gpus, 77
  - m\_gpus, 77
  - m\_nodes, 77
  - m\_path, 77
  - node, 77
  - nodes, 77
  - PBSGPUParser, 76
  - parse, 77
  - setDefault, 77
  - setPath, 77
  - size, 77
- Langmuir::Potential, 78
  - coulombD, 79
  - coulombE, 79
  - coulombH, 79
  - coulombImageD, 79
  - coulombImageE, 80
  - coulombImageH, 80
  - gaussD, 80
  - gaussE, 80
  - gaussH, 80
  - gaussImageD, 80
  - gaussImageE, 81
  - gaussImageH, 81
  - m\_world, 82
  - Potential, 79
  - precalculateArrays, 81
  - setPotentialGate, 81
  - setPotentialLinear, 81
  - setPotentialTraps, 81
  - setPotentialZero, 81
  - updateCouplingConstants, 81
- Langmuir::Random, 82
  - ~Random, 83
  - chooseNo, 83
  - chooseYes, 83



- generator01, 85
- integer, 83
- m\_seed, 85
- metropolis, 83
- metropolisWithCoupling, 84
- normal, 84
- operator<<, 84, 85
- operator>>, 85
- Random, 83
- random, 84
- range, 84
- seed, 84
- twister, 85
- Langmuir::RecombinationAgent, 86
  - energyChange, 86
  - guessProbability, 86
  - RecombinationAgent, 86
  - tryToAccept, 87
- Langmuir::Simulation, 87
  - ~Simulation, 88
  - balanceCharges, 88
  - chargeAgentCoulombInteractionQtConcurrentCPU, 88
  - chargeAgentCoulombInteractionQtConcurrentGPU, 88
  - m\_world, 89
  - nextTick, 88
  - performInjections, 88
  - performIterations, 88
  - performRecombinations, 89
  - Simulation, 88
- Langmuir::SimulationParameters, 89
  - balanceCharges, 93
  - boltzmannConstant, 93
  - coulombCarriers, 93
  - coulombGaussianSigma, 93
  - currentStep, 93
  - defectPercentage, 93
  - defectsCharge, 93
  - dielectricConstant, 93
  - drainRate, 93
  - eDrainLRate, 93
  - eDrainRRate, 93
  - eSourceLRate, 94
  - eSourceRRate, 94
  - electronPercentage, 93
  - electrostaticCutoff, 94
  - electrostaticPrefactor, 94
  - elementaryCharge, 94
  - excitonBinding, 94
  - gaussianStdev, 94
  - generationRate, 94
  - gridFactor, 94
  - gridX, 94
  - gridY, 94
  - gridZ, 94
  - hDrainLRate, 95
  - hDrainRRate, 95
  - hSourceLRate, 95
  - hSourceRRate, 95
  - holePercentage, 95
  - hoppingRange, 95
  - imageCarriers, 95
  - imageDefects, 95
  - imageTraps, 95
  - inverseKT, 95
  - iterationsPrint, 95
  - iterationsReal, 95
  - maxThreads, 96
  - okCL, 96
  - openclDeviceID, 96
  - openclThreshold, 96
  - outputChkTrapPotential, 96
  - outputCoulomb, 96
  - outputIdsOnDelete, 96
  - outputIdsOnEncounter, 96
  - outputIsOn, 96
  - outputPotential, 96
  - outputPrecision, 96
  - outputStepChk, 96
  - outputStub, 97
  - outputWidth, 97
  - outputXyz, 97
  - outputXyzD, 97
  - outputXyzE, 97
  - outputXyzH, 97
  - outputXyzMode, 97
  - outputXyzT, 97
  - permittivitySpace, 97
  - randomSeed, 97
  - recombinationRange, 97
  - recombinationRate, 97
  - seedCharges, 98
  - seedPercentage, 98
  - SimulationParameters, 93
  - simulationStart, 98
  - simulationType, 98
  - slopeZ, 98
  - sourceCoulomb, 98
  - sourceMetropolis, 98
  - sourceRate, 98
  - sourceScaleArea, 98
  - temperatureKelvin, 98
  - trapPercentage, 98
  - trapPotential, 98
  - useOpenCL, 99
  - voltageLeft, 99
  - voltageRight, 99
  - workSize, 99
  - workX, 99
  - workY, 99
  - workZ, 99
- Langmuir::SourceAgent, 99
  - chooseSite, 100
  - inject, 100
  - randomNeighborSiteID, 100

- randomSiteID, 101
- shouldTransport, 101
- SourceAgent, 100
- tryToInject, 101
- tryToSeed, 101
- validToInject, 101
- Langmuir::Tolerance, 102
  - check, 103
  - checkCriteria, 103
  - converged, 103
  - m\_converged, 103
  - m\_criteria, 104
  - m\_current, 104
  - m\_flux, 104
  - m\_previous, 104
  - m\_world, 104
  - percentChange, 103
  - setCurrent, 103
  - setPrevious, 103
  - Tolerance, 103
- Langmuir::TypedVariable
  - convert, 106
  - key, 106
  - keyValue, 107
  - m\_value, 108
  - read, 107
  - TypedVariable, 106
  - value, 107
  - write, 107
- Langmuir::TypedVariable< T >, 104
- Langmuir::Variable, 108
  - isConstant, 109
  - key, 109
  - keyValue, 109
  - m\_key, 110
  - m\_mode, 110
  - mode, 110
  - operator<<, 110
  - read, 110
  - value, 110
  - Variable, 109
  - VariableModeFlag, 109
  - write, 110
- Langmuir::World, 111
  - ~World, 116
  - atMaxCharges, 116
  - atMaxElectrons, 116
  - atMaxHoles, 116
  - chargesAreBalanced, 116
  - checkPointer, 116
  - couplingConstants, 116
  - createDrains, 116
  - createSources, 116
  - defectSiteIDs, 117
  - drains, 117
  - eDrains, 117
  - eR, 117
  - eSources, 117
  - electronDrainAgentLeft, 117
  - electronDrainAgentRight, 117
  - electronGrid, 117
  - electronSourceAgentLeft, 117
  - electronSourceAgentRight, 117
  - electrons, 117
  - electronsMinusHoles, 117
  - excitonSourceAgent, 118
  - fluxes, 118
  - hDrains, 118
  - hSources, 118
  - holeDrainAgentLeft, 118
  - holeDrainAgentRight, 118
  - holeGrid, 118
  - holeSourceAgentLeft, 118
  - holeSourceAgentRight, 118
  - holes, 118
  - holesMinusElectrons, 118
  - iR, 119
  - initialize, 118
  - keyValueParser, 119
  - logger, 119
  - m\_R1, 125
  - m\_R2, 125
  - m\_checkPointer, 122
  - m\_couplingConstants, 122
  - m\_defectSiteIDs, 122
  - m\_drains, 122
  - m\_eDrains, 122
  - m\_eR, 123
  - m\_eSources, 123
  - m\_electronDrainAgentLeft, 123
  - m\_electronDrainAgentRight, 123
  - m\_electronGrid, 123
  - m\_electronSourceAgentLeft, 123
  - m\_electronSourceAgentRight, 123
  - m\_electrons, 123
  - m\_excitonSourceAgent, 123
  - m\_fluxAgents, 123
  - m\_hDrains, 123
  - m\_hSources, 124
  - m\_holeDrainAgentLeft, 123
  - m\_holeDrainAgentRight, 124
  - m\_holeGrid, 124
  - m\_holeSourceAgentLeft, 124
  - m\_holeSourceAgentRight, 124
  - m\_holes, 124
  - m\_iR, 124
  - m\_keyValueParser, 124
  - m\_logger, 124
  - m\_maxDefects, 124
  - m\_maxElectrons, 124
  - m\_maxHoles, 124
  - m\_maxTraps, 125
  - m\_ocl, 125
  - m\_parameters, 125
  - m\_potential, 125
  - m\_rand, 125

- m\_recombinationAgent, 125
- m\_sl, 125
- m\_sources, 125
- m\_trapSiteIDs, 125
- m\_trapSitePotentials, 126
- m\_xDrains, 126
- m\_xSources, 126
- maxChargeAgents, 119
- maxChargeAgentsAndChargedDefects, 119
- maxDefects, 119
- maxElectronAgents, 119
- maxHoleAgents, 119
- maxTraps, 119
- numChargeAgents, 119
- numChargeAgentsAndChargedDefects, 119
- numDefects, 120
- numElectronAgents, 120
- numHoleAgents, 120
- numTraps, 120
- opencl, 120
- parameters, 120
- percentElectronAgents, 120
- percentHoleAgents, 120
- placeDefects, 120
- placeElectrons, 120
- placeHoles, 121
- potential, 121
- R1, 121
- R2, 121
- randomNumberGenerator, 121
- reachedChargeAgents, 121
- reachedElectronAgents, 121
- reachedHoleAgents, 121
- recombinationAgent, 121
- sl, 122
- setFluxInfo, 121
- sources, 122
- trapSiteIDs, 122
- trapSitePotentials, 122
- World, 116
- xDrains, 122
- xSources, 122
- Langmuir::XYZWriter, 126
  - m\_stream, 127
  - m\_world, 127
  - write, 127
  - writeVMDInitFile, 127
  - XYZWriter, 127
- langmuirview.cpp, 135
  - main, 135
- launchCoulombKernel1
  - Langmuir::OpenCIHelper, 73
- launchCoulombKernel2
  - Langmuir::OpenCIHelper, 73
- launchGaussKernel1
  - Langmuir::OpenCIHelper, 73
- launchGaussKernel2
  - Langmuir::OpenCIHelper, 73
- Left
  - Langmuir::Grid, 50
- lifetime
  - Langmuir::ChargeAgent, 24
- load
  - Langmuir::CheckPointer, 28
- loadDefects
  - Langmuir::CheckPointer, 28
- loadElectrons
  - Langmuir::CheckPointer, 29
- loadFluxState
  - Langmuir::CheckPointer, 29
- loadHoles
  - Langmuir::CheckPointer, 29
- loadParameters
  - Langmuir::CheckPointer, 29
- loadRandomState
  - Langmuir::CheckPointer, 29
- loadTrapPotentials
  - Langmuir::CheckPointer, 29
- loadTraps
  - Langmuir::CheckPointer, 30
- Logger
  - Langmuir::Logger, 69
- logger
  - Langmuir::World, 119
- m\_R1
  - Langmuir::World, 125
- m\_R2
  - Langmuir::World, 125
- m\_agentType
  - Langmuir::Grid, 58
- m\_agents
  - Langmuir::Grid, 58
- m\_attempts
  - Langmuir::FluxAgent, 45
- m\_carrierWriter
  - Langmuir::Logger, 70
- m\_charge
  - Langmuir::ChargeAgent, 25
- m\_checkPointer
  - Langmuir::World, 122
- m\_converged
  - Langmuir::Tolerance, 103
- m\_couplingConstants
  - Langmuir::World, 122
- m\_criteria
  - Langmuir::Tolerance, 104
- m\_current
  - Langmuir::Tolerance, 104
- m\_de
  - Langmuir::ChargeAgent, 25
- m\_defectSiteIDs
  - Langmuir::World, 122
- m\_drains
  - Langmuir::World, 122
- m\_eDrains
  - Langmuir::World, 122

- m\_eR
  - Langmuir::World, [123](#)
- m\_eSources
  - Langmuir::World, [123](#)
- m\_electronDrainAgentLeft
  - Langmuir::World, [123](#)
- m\_electronDrainAgentRight
  - Langmuir::World, [123](#)
- m\_electronGrid
  - Langmuir::World, [123](#)
- m\_electronSourceAgentLeft
  - Langmuir::World, [123](#)
- m\_electronSourceAgentRight
  - Langmuir::World, [123](#)
- m\_electrons
  - Langmuir::World, [123](#)
- m\_excitonSourceAgent
  - Langmuir::World, [123](#)
- m\_excitonWriter
  - Langmuir::Logger, [70](#)
- m\_fSite
  - Langmuir::Agent, [20](#)
- m\_face
  - Langmuir::FluxAgent, [45](#)
- m\_file
  - Langmuir::OutputStream, [75](#)
- m\_flux
  - Langmuir::Tolerance, [104](#)
- m\_fluxAgents
  - Langmuir::World, [123](#)
- m\_fluxWriter
  - Langmuir::Logger, [70](#)
- m\_gpus
  - Langmuir::PBSGPUParser, [77](#)
- m\_grid
  - Langmuir::ChargeAgent, [25](#)
  - Langmuir::FluxAgent, [45](#)
- m\_hDrains
  - Langmuir::World, [123](#)
- m\_hSources
  - Langmuir::World, [124](#)
- m\_holeDrainAgentLeft
  - Langmuir::World, [123](#)
- m\_holeDrainAgentRight
  - Langmuir::World, [124](#)
- m\_holeGrid
  - Langmuir::World, [124](#)
- m\_holeSourceAgentLeft
  - Langmuir::World, [124](#)
- m\_holeSourceAgentRight
  - Langmuir::World, [124](#)
- m\_holes
  - Langmuir::World, [124](#)
- m\_iR
  - Langmuir::World, [124](#)
- m\_image
  - Langmuir::GridImage, [61](#)
- m\_info
  - Langmuir::OutputStream, [75](#)
- m\_key
  - Langmuir::Variable, [110](#)
- m\_keyValueParser
  - Langmuir::World, [124](#)
- m\_lastAttempts
  - Langmuir::FluxAgent, [45](#)
- m\_lastStep
  - Langmuir::FluxAgent, [45](#)
- m\_lastSuccesses
  - Langmuir::FluxAgent, [45](#)
- m\_lifetime
  - Langmuir::ChargeAgent, [26](#)
- m\_logger
  - Langmuir::World, [124](#)
- m\_maxDefects
  - Langmuir::World, [124](#)
- m\_maxElectrons
  - Langmuir::World, [124](#)
- m\_maxHoles
  - Langmuir::World, [124](#)
- m\_maxTraps
  - Langmuir::World, [125](#)
- m\_mode
  - Langmuir::Variable, [110](#)
- m\_neighbors
  - Langmuir::Agent, [20](#)
- m\_nodes
  - Langmuir::PBSGPUParser, [77](#)
- m\_ocl
  - Langmuir::World, [125](#)
- m\_openCLID
  - Langmuir::ChargeAgent, [26](#)
- m\_orderedNames
  - Langmuir::KeyValueParser, [67](#)
- mPainter
  - Langmuir::GridImage, [61](#)
- m\_parameters
  - Langmuir::KeyValueParser, [67](#)
  - Langmuir::World, [125](#)
- m\_path
  - Langmuir::PBSGPUParser, [77](#)
- m\_pathlength
  - Langmuir::ChargeAgent, [26](#)
- m\_potential
  - Langmuir::FluxAgent, [46](#)
  - Langmuir::World, [125](#)
- m\_potentials
  - Langmuir::Grid, [59](#)
- m\_previous
  - Langmuir::Tolerance, [104](#)
- m\_probability
  - Langmuir::FluxAgent, [46](#)
- m\_rand
  - Langmuir::World, [125](#)
- m\_recombinationAgent
  - Langmuir::World, [125](#)
- m\_removed

- Langmuir::ChargeAgent, 26
- m\_sl
  - Langmuir::World, 125
- m\_seed
  - Langmuir::Random, 85
- m\_site
  - Langmuir::Agent, 20
- m\_sources
  - Langmuir::World, 125
- m\_specialAgentCount
  - Langmuir::Grid, 59
- m\_specialAgentReserve
  - Langmuir::Grid, 59
- m\_specialAgents
  - Langmuir::Grid, 59
- m\_stream
  - Langmuir::CarrierWriter, 21
  - Langmuir::ExcitonWriter, 39
  - Langmuir::FluxWriter, 47
  - Langmuir::XYZWriter, 127
- m\_successes
  - Langmuir::FluxAgent, 46
- m\_trapSiteIDs
  - Langmuir::World, 125
- m\_trapSitePotentials
  - Langmuir::World, 126
- m\_type
  - Langmuir::Agent, 20
- m\_value
  - Langmuir::TypedVariable, 108
- m\_variableMap
  - Langmuir::KeyValueParser, 67
- m\_volume
  - Langmuir::Grid, 59
- m\_world
  - Langmuir::Agent, 20
  - Langmuir::CarrierWriter, 21
  - Langmuir::CheckPointer, 31
  - Langmuir::ExcitonWriter, 39
  - Langmuir::FluxWriter, 47
  - Langmuir::Grid, 59
  - Langmuir::GridImage, 61
  - Langmuir::KeyValueParser, 67
  - Langmuir::Logger, 70
  - Langmuir::OpenCLHelper, 73
  - Langmuir::Potential, 82
  - Langmuir::Simulation, 89
  - Langmuir::Tolerance, 104
  - Langmuir::XYZWriter, 127
- m\_xDrains
  - Langmuir::World, 126
- m\_xSize
  - Langmuir::Grid, 59
- m\_xSources
  - Langmuir::World, 126
- m\_xyPlaneArea
  - Langmuir::Grid, 59
- m\_xyzWriter
  - Langmuir::Logger, 71
- m\_xzPlaneArea
  - Langmuir::Grid, 59
- m\_ySize
  - Langmuir::Grid, 59
- m\_yzPlaneArea
  - Langmuir::Grid, 59
- m\_zSize
  - Langmuir::Grid, 60
- main
  - langmuir.cpp, 135
  - langmuirview.cpp, 135
  - test.cpp, 142
- maxChargeAgents
  - Langmuir::World, 119
- maxChargeAgentsAndChargedDefects
  - Langmuir::World, 119
- maxDefects
  - Langmuir::World, 119
- maxElectronAgents
  - Langmuir::World, 119
- maxHoleAgents
  - Langmuir::World, 119
- maxThreads
  - Langmuir::SimulationParameters, 96
- maxTraps
  - Langmuir::World, 119
- metropolis
  - Langmuir::Random, 83
- metropolisWithCoupling
  - Langmuir::Random, 84
- mode
  - Langmuir::Variable, 110
- neighborsFace
  - Langmuir::Grid, 53
- neighborsSite
  - Langmuir::Grid, 53
- newline
  - output.cpp, 137
  - output.h, 138
- nextTick
  - Langmuir::Simulation, 88
- NoFace
  - Langmuir::Grid, 50
- node
  - Langmuir::PBSGPUParser, 77
- nodes
  - Langmuir::PBSGPUParser, 77
- normal
  - Langmuir::Random, 84
- numChargeAgents
  - Langmuir::World, 119
- numChargeAgentsAndChargedDefects
  - Langmuir::World, 119
- numDefects
  - Langmuir::World, 120
- numElectronAgents
  - Langmuir::World, 120

- numHoleAgents
  - Langmuir::World, 120
- numTraps
  - Langmuir::World, 120
- okCL
  - Langmuir::SimulationParameters, 96
- OpenCIHelper
  - Langmuir::OpenCIHelper, 72
- opengl
  - Langmuir::World, 120
- openglDeviceID
  - Langmuir::SimulationParameters, 96
- openglThreshold
  - Langmuir::SimulationParameters, 96
- openglhelper.cpp, 136
- openglhelper.h, 136
- operator<<
  - Langmuir, 14, 15
  - Langmuir::KeyValueParser, 67
  - Langmuir::Random, 84, 85
  - Langmuir::Variable, 110
- operator>>
  - Langmuir, 15
  - Langmuir::Random, 85
- otherGrid
  - Langmuir::ChargeAgent, 24
  - Langmuir::ElectronAgent, 34
  - Langmuir::HoleAgent, 62
- otherType
  - Langmuir::ChargeAgent, 25
  - Langmuir::ElectronAgent, 34
  - Langmuir::HoleAgent, 63
- output.cpp, 136
  - newline, 137
  - space, 137
- output.h, 137
  - newline, 138
  - space, 138
- outputChkTrapPotential
  - Langmuir::SimulationParameters, 96
- outputCoulomb
  - Langmuir::SimulationParameters, 96
- outputIdsOnDelete
  - Langmuir::SimulationParameters, 96
- outputIdsOnEncounter
  - Langmuir::SimulationParameters, 96
- OutputInfo
  - Langmuir::OutputInfo, 74
- outputIsOn
  - Langmuir::SimulationParameters, 96
- outputPotential
  - Langmuir::SimulationParameters, 96
- outputPrecision
  - Langmuir::SimulationParameters, 96
- outputStepChk
  - Langmuir::SimulationParameters, 96
- OutputStream
  - Langmuir::OutputStream, 75
- outputStub
  - Langmuir::SimulationParameters, 97
- outputWidth
  - Langmuir::SimulationParameters, 97
- outputXyz
  - Langmuir::SimulationParameters, 97
- outputXyzD
  - Langmuir::SimulationParameters, 97
- outputXyzE
  - Langmuir::SimulationParameters, 97
- outputXyzH
  - Langmuir::SimulationParameters, 97
- outputXyzMode
  - Langmuir::SimulationParameters, 97
- outputXyzT
  - Langmuir::SimulationParameters, 97
- PBSGPUParser
  - Langmuir::PBSGPUParser, 76
- Parameters
  - Langmuir::CheckPointer, 28
- parameters
  - Langmuir::KeyValueParser, 67
  - Langmuir::World, 120
- parameters.h, 138
- parse
  - Langmuir::KeyValueParser, 67
  - Langmuir::PBSGPUParser, 77
- pathlength
  - Langmuir::ChargeAgent, 25
- pbsgpuparser.cpp, 138
- pbsgpuparser.h, 139
- percentChange
  - Langmuir::Tolerance, 103
- percentElectronAgents
  - Langmuir::World, 120
- percentHoleAgents
  - Langmuir::World, 120
- performInjections
  - Langmuir::Simulation, 88
- performIterations
  - Langmuir::Simulation, 88
- performRecombinations
  - Langmuir::Simulation, 89
- permittivitySpace
  - Langmuir::SimulationParameters, 97
- placeDefects
  - Langmuir::World, 120
- placeElectrons
  - Langmuir::World, 120
- placeHoles
  - Langmuir::World, 121
- Potential
  - Langmuir::Potential, 79
- potential
  - Langmuir::FluxAgent, 43
  - Langmuir::Grid, 53
  - Langmuir::World, 121
- potential.cpp, 139

- potential.h, [139](#)
- precalculateArrays
  - Langmuir::Potential, [81](#)
- progress
  - langmuir.cpp, [135](#)
- R1
  - Langmuir::World, [121](#)
- R2
  - Langmuir::World, [121](#)
- README.md, [140](#)
- rand.cpp, [140](#)
- rand.h, [140](#)
- Random
  - Langmuir::Random, [83](#)
- random
  - Langmuir::Random, [84](#)
- RandomState
  - Langmuir::CheckPointer, [28](#)
- randomNeighborSiteID
  - Langmuir::SourceAgent, [100](#)
- randomNumberGenerator
  - Langmuir::World, [121](#)
- randomSeed
  - Langmuir::SimulationParameters, [97](#)
- randomSiteID
  - Langmuir::SourceAgent, [101](#)
- range
  - Langmuir::Random, [84](#)
- rate
  - Langmuir::FluxAgent, [43](#)
- reachedChargeAgents
  - Langmuir::World, [121](#)
- reachedElectronAgents
  - Langmuir::World, [121](#)
- reachedHoleAgents
  - Langmuir::World, [121](#)
- read
  - Langmuir::TypedVariable, [107](#)
  - Langmuir::Variable, [110](#)
- RecombinationAgent
  - Langmuir::RecombinationAgent, [86](#)
- recombinationAgent
  - Langmuir::World, [121](#)
- recombinationRange
  - Langmuir::SimulationParameters, [97](#)
- recombinationRate
  - Langmuir::SimulationParameters, [97](#)
- registerAgent
  - Langmuir::Grid, [54](#)
- registerDefect
  - Langmuir::Grid, [54](#)
- registerSpecialAgent
  - Langmuir::Grid, [54](#)
- registerVariable
  - Langmuir::KeyValueParser, [67](#)
- removed
  - Langmuir::ChargeAgent, [25](#)
- reportCarrier
  - Langmuir::Logger, [69](#)
- reportExciton
  - Langmuir::Logger, [69](#)
- reportFluxStream
  - Langmuir::Logger, [69](#)
- reportXYZStream
  - Langmuir::Logger, [69](#)
- resetCounters
  - Langmuir::FluxAgent, [43](#)
- Right
  - Langmuir::Grid, [50](#)
- SIZE
  - Langmuir::Agent, [18](#)
- sl
  - Langmuir::World, [122](#)
- save
  - Langmuir::CheckPointer, [30](#)
  - Langmuir::GridImage, [61](#)
  - Langmuir::KeyValueParser, [67](#)
- saveCarriersImage
  - Langmuir::Logger, [69](#)
- saveCoulombEnergy
  - Langmuir::Logger, [70](#)
- saveDefectImage
  - Langmuir::Logger, [70](#)
- saveDefects
  - Langmuir::CheckPointer, [30](#)
- saveElectronImage
  - Langmuir::Logger, [70](#)
- saveElectrons
  - Langmuir::CheckPointer, [30](#)
- saveFluxState
  - Langmuir::CheckPointer, [30](#)
- saveGridPotential
  - Langmuir::Logger, [70](#)
- saveHoleImage
  - Langmuir::Logger, [70](#)
- saveHoles
  - Langmuir::CheckPointer, [30](#)
- savelImage
  - Langmuir::Logger, [70](#)
- saveParameters
  - Langmuir::CheckPointer, [31](#)
- saveRandomState
  - Langmuir::CheckPointer, [31](#)
- saveTrapImage
  - Langmuir::Logger, [70](#)
- saveTrapPotentials
  - Langmuir::CheckPointer, [31](#)
- saveTraps
  - Langmuir::CheckPointer, [31](#)
- Section
  - Langmuir::CheckPointer, [28](#)
- seed
  - Langmuir::Random, [84](#)
- seedCharges
  - Langmuir::SimulationParameters, [98](#)
- seedPercentage

- Langmuir::SimulationParameters, 98
- setAttempts
  - Langmuir::FluxAgent, 43
- setCalculatedValues
  - Langmuir, 15
- setCurrent
  - Langmuir::Tolerance, 103
- setCurrentSite
  - Langmuir::Agent, 19
- setDefault
  - Langmuir::PBSGPUParser, 77
- setFluxInfo
  - Langmuir::World, 121
- setFutureSite
  - Langmuir::Agent, 19
- setNeighbors
  - Langmuir::Agent, 19
- setOpenCLID
  - Langmuir::ChargeAgent, 25
- setPath
  - Langmuir::PBSGPUParser, 77
- setPotential
  - Langmuir::FluxAgent, 43
  - Langmuir::Grid, 54
- setPotentialGate
  - Langmuir::Potential, 81
- setPotentialLinear
  - Langmuir::Potential, 81
- setPotentialTraps
  - Langmuir::Potential, 81
- setPotentialZero
  - Langmuir::Potential, 81
- setPrevious
  - Langmuir::Tolerance, 103
- setRate
  - Langmuir::FluxAgent, 43
- setRateSmartly
  - Langmuir::FluxAgent, 43
- setRemoved
  - Langmuir::ChargeAgent, 25
- setSuccesses
  - Langmuir::FluxAgent, 44
- shouldTransport
  - Langmuir::ExcitonSourceAgent, 38
  - Langmuir::FluxAgent, 44
  - Langmuir::SourceAgent, 101
- Simulation
  - Langmuir::Simulation, 88
- simulation.cpp, 141
- simulation.h, 141
- SimulationParameters
  - Langmuir::SimulationParameters, 93
- simulationStart
  - Langmuir::SimulationParameters, 98
- simulationType
  - Langmuir::SimulationParameters, 98
- size
  - Langmuir::PBSGPUParser, 77
- sliceIndex
  - Langmuir::Grid, 54
- slopeZ
  - Langmuir::SimulationParameters, 98
- Source
  - Langmuir::Agent, 18
- SourceAgent
  - Langmuir::SourceAgent, 100
- sourceCoulomb
  - Langmuir::SimulationParameters, 98
- sourceMetropolis
  - Langmuir::SimulationParameters, 98
- sourceRate
  - Langmuir::SimulationParameters, 98
- sourceScaleArea
  - Langmuir::SimulationParameters, 98
- sourceagent.cpp, 141
- sourceagent.h, 142
- sources
  - Langmuir::World, 122
- space
  - output.cpp, 137
  - output.h, 138
- specialAgentCount
  - Langmuir::Grid, 55
- stepsSinceLast
  - Langmuir::FluxAgent, 44
- storeLast
  - Langmuir::FluxAgent, 44
- successProbability
  - Langmuir::FluxAgent, 45
- successProbabilitySinceLast
  - Langmuir::FluxAgent, 45
- successRate
  - Langmuir::FluxAgent, 45
- successRateSinceLast
  - Langmuir::FluxAgent, 45
- successes
  - Langmuir::FluxAgent, 44
- successesSinceLast
  - Langmuir::FluxAgent, 44
- temperatureKelvin
  - Langmuir::SimulationParameters, 98
- test.cpp, 142
  - main, 142
- toQString
  - Langmuir::Agent, 19
  - Langmuir::Grid, 55
- toggleOpenCL
  - Langmuir::OpenCLHelper, 73
- Tolerance
  - Langmuir::Tolerance, 103
- tolerance.cpp, 142
- tolerance.h, 143
- Top
  - Langmuir::Grid, 50
- totalDistance
  - Langmuir::Grid, 55



- TrapPotentials
  - Langmuir::CheckPointer, 28
- trapPercentage
  - Langmuir::SimulationParameters, 98
- trapPotential
  - Langmuir::SimulationParameters, 98
- trapPotentials
  - Langmuir::ConfigurationInfo, 32
- trapSiteIDs
  - Langmuir::World, 122
- trapSitePotentials
  - Langmuir::World, 122
- Traps
  - Langmuir::CheckPointer, 28
- traps
  - Langmuir::ConfigurationInfo, 32
- tryToAccept
  - Langmuir::DrainAgent, 33
  - Langmuir::RecombinationAgent, 87
- tryToInject
  - Langmuir::SourceAgent, 101
- tryToSeed
  - Langmuir::SourceAgent, 101
- twister
  - Langmuir::Random, 85
- Type
  - Langmuir::Agent, 18
- TypedVariable
  - Langmuir::TypedVariable, 106
- unregisterAgent
  - Langmuir::Grid, 55
- unregisterDefect
  - Langmuir::Grid, 55
- unregisterSpecialAgent
  - Langmuir::Grid, 55
- updateCouplingConstants
  - Langmuir::Potential, 81
- useOpenCL
  - Langmuir::SimulationParameters, 99
- validToInject
  - Langmuir::ElectronSourceAgent, 37
  - Langmuir::ExcitonSourceAgent, 38
  - Langmuir::HoleSourceAgent, 65
  - Langmuir::SourceAgent, 101
- value
  - Langmuir::TypedVariable, 107
  - Langmuir::Variable, 110
- Variable
  - Langmuir::Variable, 109
- variable.h, 143
- VariableModeFlag
  - Langmuir::Variable, 109
- voltageLeft
  - Langmuir::SimulationParameters, 99
- voltageRight
  - Langmuir::SimulationParameters, 99
- volume
  - Langmuir::Grid, 55
- work
  - copy\_to\_frank, 11
- workSize
  - Langmuir::SimulationParameters, 99
- workX
  - Langmuir::SimulationParameters, 99
- workY
  - Langmuir::SimulationParameters, 99
- workZ
  - Langmuir::SimulationParameters, 99
- World
  - Langmuir::World, 116
- world.cpp, 144
- world.h, 144
- write
  - Langmuir::CarrierWriter, 21
  - Langmuir::ExcitonWriter, 39
  - Langmuir::FluxWriter, 47
  - Langmuir::TypedVariable, 107
  - Langmuir::Variable, 110
  - Langmuir::XYZWriter, 127
- writeVMDInitFile
  - Langmuir::XYZWriter, 127
- writer.cpp, 145
- writer.h, 145
- xDistance
  - Langmuir::Grid, 55
- xDistancei
  - Langmuir::Grid, 56
- xDrains
  - Langmuir::World, 122
- xImageDistance
  - Langmuir::Grid, 56
- xImageDistancei
  - Langmuir::Grid, 56
- xSize
  - Langmuir::Grid, 56
- xSources
  - Langmuir::World, 122
- XYZWriter
  - Langmuir::XYZWriter, 127
- xyPlaneArea
  - Langmuir::Grid, 56
- yDistance
  - Langmuir::Grid, 56
- yDistancei
  - Langmuir::Grid, 57
- yImageDistance
  - Langmuir::Grid, 57
- yImageDistancei
  - Langmuir::Grid, 57
- ySize
  - Langmuir::Grid, 57
- zDistance

Langmuir::Grid, [57](#)  
zDistancei  
    Langmuir::Grid, [57](#)  
zImageDistance  
    Langmuir::Grid, [58](#)  
zImageDistancei  
    Langmuir::Grid, [58](#)  
zSize  
    Langmuir::Grid, [58](#)