Circles

0.2

Generated by Doxygen 1.8.10

Wed Jan 27 2016 19:19:31

Contents

1 README 1

2 Namespace Index 3

2.1 Namespace List . . . 3

3 Hierarchical Index 5

3.1 Class Hierarchy . . . 5

4 Class Index 7

4.1 Class List . . . 7

5 File Index 9

5.1 File List . . . 9

6 Namespace Documentation 11

6.1 Circles Namespace Reference . . . 11

6.2 Circles::Graph Namespace Reference . . . 11

6.2.1 Typedef Documentation . . . 11

6.2.1.1 Node . . . 11

6.2.2 Function Documentation . . . 11

6.2.2.1 operator==(const Edge &lhs, const Edge &rhs) . . . 11

6.3 Circles::Packing Namespace Reference . . . 11

6.3.1 Function Documentation . . . 12

6.3.1.1 operator<(const Circle &lhs, const Circle &rhs) . . . 12

6.3.1.2 operator==(const EuclidCircle &lhs, const EuclidCircle &rhs) . . . 12

6.3.1.3 operator==(const HyperCircle &lhs, const HyperCircle &rhs) . . . 12

6.3.1.4 operator==(const EuclidPacking &lhs, const EuclidPacking &rhs) . . . 12

6.4 Ui Namespace Reference . . . 12

7 Class Documentation 13

7.1 Boundary Class Reference . . . 13

7.1.1 Constructor & Destructor Documentation . . . 13

7.1.1.1 Boundary() . . . 13

7.1.2 Member Function Documentation . . . 13

7.1.2.1 boundingRect() const override . . . 13
7.1.2.2 paint(QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget=0) override . . . 13
7.2 Circle Class Reference . . . 13
7.2.1 Detailed Description . . . 14
7.2.2 Member Enumeration Documentation . . . 14
7.2.2.1 SelectionState . . . 14
7.2.3 Constructor & Destructor Documentation . . . 14
7.2.3.1 Circle(Node *n, Packing *p) . . . 14
7.2.4 Member Function Documentation . . . 14
7.2.4.1 boundingRect() const Q_DECL_OVERRIDE . . . 14
7.2.4.2 getNode() . . . 14
7.2.4.3 getSelectionState() . . . 14
7.2.4.4 paint(QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget=nullptr) Q_DECL_OVERRIDE . . . 14
7.2.4.5 setSelectionState(SelectionState s) . . . 15
7.2.4.6 shape() const Q_DECL_OVERRIDE . . . 15
7.3 Circles::Packing::Circle Class Reference . . . 15
7.3.1 Constructor & Destructor Documentation . . . 15
7.3.1.1 Circle() . . . 15
7.3.2 Member Function Documentation . . . 15
7.3.2.1 center() const =0 . . . 15
7.3.2.2 index() const . . . 16
7.3.2.3 projCenter() const =0 . . . 16
7.3.2.4 projRadius() const =0 . . . 16
7.3.2.5 radius() const =0 . . . 16
7.3.2.6 setCenter(QPointF c)=0 . . . 16
7.3.2.7 setIndex(int index) . . . 17
7.3.2.8 setRadius(qreal r)=0 . . . 17
7.3.3 Friends And Related Function Documentation . . . 17
7.3.3.1 operator< . . . 17
7.3.4 Member Data Documentation . . . 17
7.3.4.1 _center . . . 17
7.3.4.2 _index . . . 17
7.3.4.3 _radius . . . 17
7.4 Connector Class Reference . . . 18
7.4.1 Constructor & Destructor Documentation . . . 18
7.4.1.1 Connector(Node *n1, Node *n2) . . . 18
7.4.2 Member Function Documentation . . . 18
7.4.2.1 boundingRect() const Q_DECL_OVERRIDE . . . 18

7.4.2.2 paint(QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget=nullptr) Q_DECL_OVERRIDE . . . 18
7.5 Circles::Graph::Edge Class Reference . . . 18
7.5.1 Detailed Description . . . 19
7.5.2 Constructor & Destructor Documentation . . . 19
7.5.2.1 Edge() . . . 19
7.5.2.2 Edge(int x, int y) . . . 19
7.5.2.3 Edge(const Edge &other) . . . 19
7.5.3 Member Function Documentation . . . 19
7.5.3.1 getX() . . . 19
7.5.3.2 getY() . . . 19
7.5.3.3 operator=(const Edge &other) . . . 19
7.5.3.4 set(int x, int y) . . . 19
7.5.3.5 setX(int x) . . . 19
7.5.3.6 setY(int y) . . . 19
7.5.4 Friends And Related Function Documentation . . . 19
7.5.4.1 operator== . . . 19
7.6 Circles::Packing::EuclidCircle Class Reference . . . 19
7.6.1 Detailed Description . . . 20
7.6.2 Constructor & Destructor Documentation . . . 20
7.6.2.1 EuclidCircle() . . . 20
7.6.2.2 EuclidCircle(const QPointF &center, qreal radius, int index) . . . 20
7.6.2.3 EuclidCircle(const EuclidCircle &other) . . . 20
7.6.3 Member Function Documentation . . . 20
7.6.3.1 center() const override . . . 21
7.6.3.2 operator=(const EuclidCircle &other) . . . 21
7.6.3.3 projCenter() const override . . . 21
7.6.3.4 projRadius() const override . . . 21
7.6.3.5 radius() const override . . . 21
7.6.3.6 setCenter(QPointF c) override . . . 21
7.6.3.7 setRadius(qreal r) override . . . 22
7.6.4 Friends And Related Function Documentation . . . 23
7.6.4.1 operator== . . . 23
7.7 Circles::Packing::EuclidPacking Class Reference . . . 23
7.7.1 Detailed Description . . . 23
7.7.2 Constructor & Destructor Documentation . . . 23
7.7.2.1 EuclidPacking() . . . 23
7.7.2.2 EuclidPacking(std::shared_ptr< Graph::Graph > g) . . . 24
7.7.2.3 EuclidPacking(std::shared_ptr< Graph::Graph > g, const QList< Circle * > &circles) . . . 25

7.7.2.4 EuclidPacking(const EuclidPacking &other) . . . 25
7.7.2.5 EuclidPacking(EuclidPacking &&other) . . . 25
7.7.3 Member Function Documentation . . . 25
7.7.3.1 operator=(const EuclidPacking &other) . . . 25
7.7.3.2 operator=(EuclidPacking &&other) . . . 25
7.7.4 Friends And Related Function Documentation . . . 25
7.7.4.1 operator== . . . 25
7.8 Circles::Graph::Graph Class Reference . . . 25
7.8.1 Detailed Description . . . 26
7.8.2 Constructor & Destructor Documentation . . . 26
7.8.2.1 Graph() . . . 26
7.8.2.2 Graph(const Graph &other) . . . 26
7.8.2.3 Graph(Graph &&other) . . . 26
7.8.3 Member Function Documentation . . . 26
7.8.3.1 addEdge(Node x, Node y) . . . 26
7.8.3.2 addEdge(Edge e) . . . 26
7.8.3.3 boundary() . . . 26
7.8.3.4 getEdges() const . . . 26
7.8.3.5 getNodes() const . . . 27
7.8.3.6 hasEdge(Node x, Node y) const . . . 27
7.8.3.7 hasFullFlower(Node n) . . . 27
7.8.3.8 isBoundary(Node n) . . . 27
7.8.3.9 neighbours(Node i) const . . . 27
7.8.3.10 operator=(const Graph &other) . . . 28
7.8.3.11 operator=(Graph &&other) . . . 28
7.8.3.12 removeEdge(Node x, Node y) . . . 28
7.8.3.13 sortedNeighbours(Node n) . . . 28
7.9 Circles::Packing::HyperCircle Class Reference . . . 28
7.9.1 Detailed Description . . . 29
7.9.2 Constructor & Destructor Documentation . . . 29
7.9.2.1 HyperCircle() . . . 29
7.9.2.2 HyperCircle(const QPointF &center, qreal radius, int index) . . . 29
7.9.2.3 HyperCircle(const HyperCircle &other) . . . 29
7.9.3 Member Function Documentation . . . 29
7.9.3.1 center() const override . . . 29
7.9.3.2 operator=(const HyperCircle &other) . . . 29
7.9.3.3 projCenter() const override . . . 29
7.9.3.4 projRadius() const override . . . 30
7.9.3.5 radius() const override . . . 30
7.9.3.6 setCenter(QPointF c) override . . . 30

7.9.3.7 setRadius(qreal r) override . . . 30
7.9.4 Friends And Related Function Documentation . . . 30
7.9.4.1 operator== . . . 30
7.10 MainWindow Class Reference . . . 31
7.10.1 Constructor & Destructor Documentation . . . 31
7.10.1.1 MainWindow(QWidget *parent=0) . . . 31
7.10.1.2 ~MainWindow() . . . 31
7.11 Node Class Reference . . . 31
7.11.1 Detailed Description . . . 32
7.11.2 Constructor & Destructor Documentation . . . 32
7.11.2.1 Node(const Node *n) . . . 32
7.11.2.2 Node(int id) . . . 32
7.11.2.3 Node(int id, const QPointF &position, qreal radius=0) . . . 32
7.11.2.4 ~Node() . . . 33
7.11.3 Member Function Documentation . . . 33
7.11.3.1 addNeibhour(Node *node) . . . 33
7.11.3.2 delNeibhour(Node *node) . . . 33
7.11.3.3 delPosition() . . . 33
7.11.3.4 generateHexArray(const QRectF &area, qreal radius) . . . 33
7.11.3.5 generateHexArray(const QPointF &startpos, int w, int h, qreal radius) . . . 33
7.11.3.6 getColor() . . . 33
7.11.3.7 getId() . . . 33
7.11.3.8 getNeibhourCount() . . . 33
7.11.3.9 getNeibhours() . . . 33
7.11.3.10 getPosition() . . . 33
7.11.3.11 getRadius() . . . 33
7.11.3.12 hasFullFlower() . . . 33
7.11.3.13 hasPosition() . . . 34
7.11.3.14 isNeibhour(Node *node) . . . 34
7.11.3.15 purgeNeibhours() . . . 34
7.11.3.16 setColor(const QColor &value) . . . 34
7.11.3.17 setId(int value) . . . 34
7.11.3.18 setPosition(const QPointF &position) . . . 34
7.11.3.19 setRadius(const qreal value) . . . 34
7.11.3.20 sortNeibhours() . . . 34
7.11.4 Member Data Documentation . . . 34
7.11.4.1 bHasPosition . . . 34
7.11.4.2 color . . . 34
7.11.4.3 id . . . 34
7.11.4.4 neibhours . . . 34

7.11.4.5 position . . . 34
7.11.4.6 radius . . . 34
7.11.4.7 sortedNeibhours . . . 34
7.12 Packing Class Reference . . . 34
7.12.1 Detailed Description . . . 36
7.12.2 Constructor & Destructor Documentation . . . 36
7.12.2.1 Packing(const Packing *p) . . . 36
7.12.2.2 Packing(PackingType type=PackingType::EuclideanPacking) . . . 36
7.12.2.3 Packing(QList< Node * > nodes, PackingType type=PackingType::Euclidean↩ Packing) . . . 36
7.12.2.4 ~Packing() . . . 37
7.12.3 Member Function Documentation . . . 37
7.12.3.1 addCircle(Node *n) . . . 37
7.12.3.2 addNode(Node *n) . . . 37
7.12.3.3 addNode_fast(Node *n) . . . 37
7.12.3.4 angle(Node *r, Node *ra, Node *rb) . . . 37
7.12.3.5 angle_euclidean(Node *r, Node *ra, Node *rb) . . . 37
7.12.3.6 angle_hyperbolic(Node *r, Node *ra, Node *rb) . . . 37
7.12.3.7 anglesum(Node *r) . . . 37
7.12.3.8 delNode(Node *n) . . . 38
7.12.3.9 delNode_fast(Node *n) . . . 38
7.12.3.10 drawForeground(QPainter *painter, const QRectF &rect) Q_DECL_OVERRIDE . . . 38
7.12.3.11 getDrawCenters() . . . 38
7.12.3.12 getDrawCircles() . . . 38
7.12.3.13 getDrawIndicies() . . . 38
7.12.3.14 getDrawLinks() . . . 38
7.12.3.15 getNodes() . . . 38
7.12.3.16 getType() . . . 38
7.12.3.17 isExterior(Node *n) . . . 38
7.12.3.18 isInterior(Node *n) . . . 38
7.12.3.19 layout . . . 38
7.12.3.20 layout_euclidean(int centerCircle) . . . 39
7.12.3.21 layout_hyperbolic(int centerCircle) . . . 39
7.12.3.22 mousePressEvent(QGraphicsSceneMouseEvent *mouseEvent) Q_DECL_OV↩ ERRIDE . . . 39
7.12.3.23 newNodeSelected . . . 39
7.12.3.24 purgeCircles() . . . 39
7.12.3.25 recomputeConnectors() . . . 39
7.12.3.26 refreshCircles() . . . 39
7.12.3.27 repack . . . 39

7.12.3.28 resetIds() . . . 39
7.12.3.29 setDrawBoundary . . . 39
7.12.3.30 setDrawCenters . . . 39
7.12.3.31 setDrawCircles . . . 39
7.12.3.32 setDrawIndicies . . . 39
7.12.3.33 setDrawLinks . . . 39
7.12.3.34 setPackingType(PackingType type) . . . 39
7.12.4 Member Data Documentation . . . 40
7.12.4.1 boundaryNodes . . . 40
7.12.4.2 centerCircleID . . . 40
7.12.4.3 circles . . . 40
7.12.4.4 connectors . . . 40
7.12.4.5 drawCenters . . . 40
7.12.4.6 drawCircles . . . 40
7.12.4.7 drawIndicies . . . 40
7.12.4.8 drawLinks . . . 40
7.12.4.9 nodes . . . 40
7.12.4.10 selectedCircle . . . 40
7.12.4.11 type . . . 40
7.13 Circles::Packing::Packing Class Reference . . . 40
7.13.1 Detailed Description . . . 41
7.13.2 Member Function Documentation . . . 41
7.13.2.1 angle(const QPointF &p, const QPointF &p1, const QPointF &p2) const =0 . . . 41
7.13.2.2 anglesum(const Circle &c) const . . . 41
7.13.2.3 circle(int index) const . . . 41
7.13.2.4 circles() const . . . 42
7.13.2.5 graph() const . . . 42
7.13.2.6 layout(int centerNode)=0 . . . 42
7.13.2.7 repack(qreal epsilon, qreal outerRadius)=0 . . . 42
7.13.3 Member Data Documentation . . . 42
7.13.3.1 _circles . . . 42
7.13.3.2 _graph . . . 42
7.14 PackingView Class Reference . . . 43
7.14.1 Constructor & Destructor Documentation . . . 43
7.14.1.1 PackingView(QWidget *parent=0) . . . 43
7.14.1.2 PackingView(Packing *p, QWidget *parent=0) . . . 43
7.14.1.3 ~PackingView() . . . 43
7.14.2 Member Function Documentation . . . 43
7.14.2.1 setPacking . . . 43
7.15 PFile Class Reference . . . 43

7.15.1 Constructor & Destructor Documentation . . . 44
7.15.1.1 PFile(QString filename) . . . 44
7.15.2 Member Function Documentation . . . 44
7.15.2.1 generatePacking() . . . 44
7.16 PropertyWindow Class Reference . . . 44
7.16.1 Constructor & Destructor Documentation . . . 44
7.16.1.1 PropertyWindow(QWidget *parent=0) . . . 44
7.16.1.2 ~PropertyWindow() . . . 44
7.16.2 Member Function Documentation . . . 44
7.16.2.1 refresh . . . 44
7.16.2.2 setNode . . . 44
7.17 SelectionPacking Class Reference . . . 45
7.17.1 Member Enumeration Documentation . . . 45
7.17.1.1 MouseMode . . . 45
7.17.2 Constructor & Destructor Documentation . . . 46
7.17.2.1 SelectionPacking(PackingType type=PackingType::EuclideanPacking) . . . 46
7.17.2.2 SelectionPacking(QList< Node * > nodes, PackingType type=PackingType::HyperbolicPacking) . . . 46
7.17.3 Member Function Documentation . . . 46
7.17.3.1 addToSelection(Circle *c) . . . 46
7.17.3.2 clearSelection() . . . 46
7.17.3.3 isInSelection(Circle *c) . . . 46
7.17.3.4 mouseMoveEvent(QGraphicsSceneMouseEvent *event) Q_DECL_OVERRIDE . . . 46
7.17.3.5 mousePressEvent(QGraphicsSceneMouseEvent *mouseEvent) Q_DECL_OVERRIDE . . . 46
7.17.3.6 mouseReleaseEvent(QGraphicsSceneMouseEvent *event) Q_DECL_OVERRIDE . . . 46
7.17.3.7 removeFromSelection(Circle *c) . . . 46
7.18 SelectionVertex Class Reference . . . 46
7.18.1 Detailed Description . . . 47
7.18.2 Constructor & Destructor Documentation . . . 47
7.18.2.1 SelectionVertex(QPointF position, QColor color=QColor(255, 0, 0)) . . . 47
7.18.3 Member Function Documentation . . . 47
7.18.3.1 boundingRect() const Q_DECL_OVERRIDE . . . 47
7.18.3.2 paint(QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) Q_DECL_OVERRIDE . . . 47
7.18.4 Member Data Documentation . . . 47
7.18.4.1 size . . . 47
7.18.4.2 thickness . . . 47
7.19 ShapeSelector Class Reference . . . 47
7.19.1 Constructor & Destructor Documentation . . . 48

7.19.1.1 ShapeSelector(QWidget *parent=0) . . . 48
7.19.1.2 ~ShapeSelector() . . . 48
7.19.2 Member Function Documentation . . . 48
7.19.2.1 packingAccepted . . . 48
7.20 ShapeSelectorGraphicsView Class Reference . . . 48
7.20.1 Constructor & Destructor Documentation . . . 49
7.20.1.1 ShapeSelectorGraphicsView(QWidget *parent=nullptr) . . . 49
7.20.1.2 ShapeSelectorGraphicsView(QGraphicsScene *scene, QWidget *parent=nullptr) 49
7.20.2 Member Function Documentation . . . 49
7.20.2.1 gotClick . . . 49
7.20.2.2 hasHeightForWidth() const . . . 49
7.20.2.3 heightForWidth(int w) const . . . 49
7.20.2.4 mousePressEvent(QMouseEvent *event) . . . 49
7.20.2.5 resizeEvent(QResizeEvent *event) . . . 49

8 File Documentation **51**

8.1 graph/Edge.cpp File Reference . . . 51
8.2 graph/Edge.hpp File Reference . . . 51
8.3 graph/Graph.cpp File Reference . . . 51
8.4 graph/Graph.hpp File Reference . . . 51
8.5 graphics/Boundary.cpp File Reference . . . 52
8.5.1 Macro Definition Documentation . . . 52
8.5.1.1 PI . . . 52
8.6 graphics/Boundary.hpp File Reference . . . 52
8.7 graphics/Circle.cpp File Reference . . . 52
8.8 packing/Circle.cpp File Reference . . . 53
8.9 graphics/Circle.hpp File Reference . . . 53
8.10 packing/Circle.hpp File Reference . . . 53
8.11 graphics/Connector.cpp File Reference . . . 53
8.12 graphics/Connector.hpp File Reference . . . 54
8.13 graphics/Packing.cpp File Reference . . . 54
8.13.1 Macro Definition Documentation . . . 54
8.13.1.1 PI . . . 54
8.14 packing/Packing.cpp File Reference . . . 54
8.15 graphics/Packing.hpp File Reference . . . 54
8.15.1 Enumeration Type Documentation . . . 55
8.15.1.1 PackingType . . . 55
8.16 packing/Packing.hpp File Reference . . . 55
8.17 graphics/SelectionPacking.cpp File Reference . . . 55
8.18 graphics/SelectionPacking.hpp File Reference . . . 56

8.19 graphics/SelectionVertex.cpp File Reference . . . 56
8.20 graphics/SelectionVertex.hpp File Reference . . . 56
8.21 graphics/ShapeSelectorGraphicsView.cpp File Reference . . . 56
8.22 graphics/ShapeSelectorGraphicsView.hpp File Reference . . . 56
8.23 main.cpp File Reference . . . 57
8.23.1 Function Documentation . . . 57
8.23.1.1 main(int argc, char ∗argv[]) . . . 57
8.24 Node.cpp File Reference . . . 57
8.24.1 Macro Definition Documentation . . . 57
8.24.1.1 PI . . . 57
8.25 Node.hpp File Reference . . . 57
8.26 packing/EuclidCircle.cpp File Reference . . . 57
8.27 packing/EuclidCircle.hpp File Reference . . . 58
8.28 packing/EuclidPacking.cpp File Reference . . . 58
8.29 packing/EuclidPacking.hpp File Reference . . . 58
8.30 packing/HyperCircle.cpp File Reference . . . 58
8.31 packing/HyperCircle.hpp File Reference . . . 59
8.32 PFile.cpp File Reference . . . 59
8.33 PFile.hpp File Reference . . . 59
8.34 README.md File Reference . . . 59
8.35 ui/MainWindow.cpp File Reference . . . 60
8.35.1 Macro Definition Documentation . . . 60
8.35.1.1 PI . . . 60
8.36 ui/MainWindow.hpp File Reference . . . 60
8.37 ui/PackingView.cpp File Reference . . . 60
8.38 ui/PackingView.hpp File Reference . . . 61
8.39 ui/PropertyWindow.cpp File Reference . . . 61
8.40 ui/PropertyWindow.hpp File Reference . . . 61
8.41 ui/ShapeSelector.cpp File Reference . . . 62
8.42 ui/ShapeSelector.hpp File Reference . . . 62

Index **63**

Chapter 1

README

Circles Circles is a program which interacts with circle packings. This program is designed to aid in the exploration of approximation of Riemann mappings from a polygon to the complex unit disc.

The program is not currently feature-complete.

Above is a screenshot of Circles in action, performing a repack of the 'owl' packing.

Features

- Supports circle packings in both standard euclidean and poincare-disc hyperbolic geometries
- Support reading of p-files generated with `Ken Stephenson's CirclePack tool`
- Toggable view of circles, center marks, indicies, tangency lines, and disc boundary
- Properties explorer to show properties of any circle
- Supports extraction of geometric shapes from hexagon-tilings for repacking

Building

This program was built and tested with QT 5.5.1 using the MINGW4.9.2 32-bit kit on Windows. Compilation shoudl also succeed with the msvc2013 kit, or any kit that supports C++14 features.

Compatibility with other QT-compatible compilers is not guaranteed, but should work just fine.

This program is still a work in progress.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Circles 11
Circles::Graph 11
Circles::Packing 11
Ui 12

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Circles::Packing::Circle . . . 15
  Circles::Packing::EuclidCircle . . . 19
  Circles::Packing::HyperCircle . . . 28
Circles::Graph::Edge . . . 18
Circles::Graph::Graph . . . 25
Node . . . 31
Circles::Packing::Packing . . . 40
  Circles::Packing::EuclidPacking . . . 23
PFile . . . 43
QDialog
  PropertyWindow . . . 44
  ShapeSelector . . . 47
QGraphicsItem
  Boundary . . . 13
  Circle . . . 13
  Connector . . . 18
  SelectionVertex . . . 46
QGraphicsScene
  Packing . . . 34
    SelectionPacking . . . 45
QGraphicsView
  ShapeSelectorGraphicsView . . . 48
QMainWindow
  MainWindow . . . 31
QWidget
  PackingView . . . 43

Chapter 4

Class Index

4.1 Class List

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

Boundary . . . 13
Circle
    Abstract circle. A circle may exist in either a hyperbolic or euclidean geometry . . . 13
Circles::Packing::Circle . . . 15
Connector . . . 18
Circles::Graph::Edge . . . 18
Circles::Packing::EuclidCircle . . . 19
Circles::Packing::EuclidPacking . . . 23
Circles::Graph::Graph . . . 25
Circles::Packing::HyperCircle . . . 28
MainWindow . . . 31
Node
    Metadata of a circle in a circle packing . . . 31
Packing
    Abstract class which defines a circle packing. A circle packing may be either euclidean or hyperbolic, as found in EuclideanPacking and HyperbolicPacking . . . 34
Circles::Packing::Packing . . . 40
PackingView . . . 43
PFile . . . 43
PropertyWindow . . . 44
SelectionPacking . . . 45
SelectionVertex . . . 46
ShapeSelector . . . 47
ShapeSelectorGraphicsView . . . 48

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

main.cpp . . . 57
Node.cpp . . . 57
Node.hpp . . . 57
PFile.cpp . . . 59
PFile.hpp . . . 59
graph/Edge.cpp . . . 51
graph/Edge.hpp . . . 51
graph/Graph.cpp . . . 51
graph/Graph.hpp . . . 51
graphics/Boundary.cpp . . . 52
graphics/Boundary.hpp . . . 52
graphics/Circle.cpp . . . 52
graphics/Circle.hpp . . . 53
graphics/Connector.cpp . . . 53
graphics/Connector.hpp . . . 54
graphics/Packing.cpp . . . 54
graphics/Packing.hpp . . . 54
graphics/SelectionPacking.cpp . . . 55
graphics/SelectionPacking.hpp . . . 56
graphics/SelectionVertex.cpp . . . 56
graphics/SelectionVertex.hpp . . . 56
graphics/ShapeSelectorGraphicsView.cpp . . . 56
graphics/ShapeSelectorGraphicsView.hpp . . . 56
packing/Circle.cpp . . . 53
packing/Circle.hpp . . . 53
packing/EuclidCircle.cpp . . . 57
packing/EuclidCircle.hpp . . . 58
packing/EuclidPacking.cpp . . . 58
packing/EuclidPacking.hpp . . . 58
packing/HyperCircle.cpp . . . 58
packing/HyperCircle.hpp . . . 59
packing/Packing.cpp . . . 54
packing/Packing.hpp . . . 55
ui/MainWindow.cpp . . . 60
ui/MainWindow.hpp . . . 60
ui/PackingView.cpp . . . 60
ui/PackingView.hpp . . . 61
ui/PropertyWindow.cpp . . . 61

ui/PropertyWindow.hpp 61
ui/ShapeSelector.cpp 62
ui/ShapeSelector.hpp 62

Chapter 6

Namespace Documentation

6.1 Circles Namespace Reference

Namespaces

- Graph
- Packing

6.2 Circles::Graph Namespace Reference

Classes

- class Edge
- class Graph

Typedefs

- typedef int Node

Functions

- bool operator== (const Edge &lhs, const Edge &rhs)

6.2.1 Typedef Documentation

6.2.1.1 typedef int Circles::Graph::Node

6.2.2 Function Documentation

6.2.2.1 bool Circles::Graph::operator== (const Edge & *lhs,* const Edge & *rhs*)

6.3 Circles::Packing Namespace Reference

Classes

- class Circle

- class EuclidCircle
- class EuclidPacking
- class HyperCircle
- class Packing

Functions

- bool operator< (const Circle &lhs, const Circle &rhs)
- bool operator== (const EuclidCircle &lhs, const EuclidCircle &rhs)
- bool operator== (const EuclidPacking &lhs, const EuclidPacking &rhs)
- bool operator== (const HyperCircle &lhs, const HyperCircle &rhs)

6.3.1 Function Documentation

6.3.1.1 bool Circles::Packing::operator< (const Circle & *lhs,* const Circle & *rhs*)

Order comparison for sorting in lists, etc.

Parameters

lhs	
rhs	

Returns

6.3.1.2 bool Circles::Packing::operator== (const EuclidCircle & *lhs,* const EuclidCircle & *rhs*)

6.3.1.3 bool Circles::Packing::operator== (const HyperCircle & *lhs,* const HyperCircle & *rhs*)

6.3.1.4 bool Circles::Packing::operator== (const EuclidPacking & *lhs,* const EuclidPacking & *rhs*)

6.4 Ui Namespace Reference

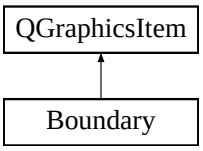
Chapter 7

Class Documentation

7.1 Boundary Class Reference

`#include <Boundary.hpp>`

Inheritance diagram for Boundary:



Public Member Functions

- Boundary ()
- QRectF boundingRect () const override
- void paint (QPainter ∗painter, const QStyleOptionGraphicsItem ∗option, QWidget ∗widget=0) override

7.1.1 Constructor & Destructor Documentation

7.1.1.1 Boundary::Boundary ( )

7.1.2 Member Function Documentation

7.1.2.1 QRectF Boundary::boundingRect ( ) const [override]

7.1.2.2 void Boundary::paint (QPainter ∗ *painter,* const QStyleOptionGraphicsItem ∗ *option,* QWidget ∗ *widget* = 0) [override]

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

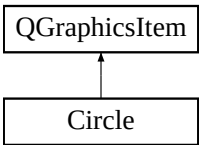
- graphics/Boundary.hpp
- graphics/Boundary.cpp

7.2 Circle Class Reference

The Circle class represents an abstract circle. A circle may exist in either a hyperbolic or euclidean geometry.

`#include <Circle.hpp>`

Inheritance diagram for Circle:



Public Types

- enum SelectionState { SelectionState::None, SelectionState::Selected, SelectionState::Surrounded }

Public Member Functions

- Circle (Node *n, Packing *p)
- QRectF boundingRect () const Q_DECL_OVERRIDE
- QPainterPath shape () const Q_DECL_OVERRIDE
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget=nullptr) Q_DEC↩ L_OVERRIDE
- SelectionState getSelectionState ()
- void setSelectionState (SelectionState s)
- Node * getNode ()

7.2.1 Detailed Description

The Circle class represents an abstract circle. A circle may exist in either a hyperbolic or euclidean geometry.

7.2.2 Member Enumeration Documentation

7.2.2.1 enum Circle::SelectionState [strong]

Enumerator

None

Selected

Surrounded

7.2.3 Constructor & Destructor Documentation

7.2.3.1 Circle::Circle (Node * *n*, Packing * *p*)

7.2.4 Member Function Documentation

7.2.4.1 QRectF Circle::boundingRect ( ) const

7.2.4.2 Node * Circle::getNode ( )

7.2.4.3 Circle::SelectionState Circle::getSelectionState ( )

7.2.4.4 void Circle::paint (QPainter * *painter*, const QStyleOptionGraphicsItem * *option*, QWidget * *widget* = `nullptr`)

7.2.4.5 void Circle::setSelectionState (Circle::SelectionState *s*)

7.2.4.6 QPainterPath Circle::shape ( ) const

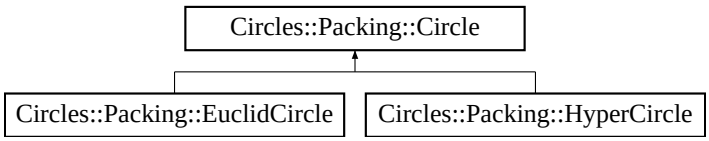
The documentation for this class was generated from the following files:

- graphics/Circle.hpp
- graphics/Circle.cpp

7.3 Circles::Packing::Circle Class Reference

`#include <Circle.hpp>`

Inheritance diagram for Circles::Packing::Circle:



Public Member Functions

- Circle ()
- int index () const
- void setIndex (int index)
- virtual QPointF center () const =0
- virtual qreal radius () const =0
- virtual QPointF projCenter () const =0
- virtual qreal projRadius () const =0
- virtual void setRadius (qreal r)=0
- virtual bool setCenter (QPointF c)=0

Protected Attributes

- int _index
- qreal _radius
- QPointF _center

Friends

- bool operator< (const Circle &lhs, const Circle &rhs)

7.3.1 Constructor & Destructor Documentation

7.3.1.1 Circle::Circle ( )

7.3.2 Member Function Documentation

7.3.2.1 virtual QPointF Circles::Packing::Circle::center ( ) const [pure virtual]

The center of the circle in it's local space.

Returns

The point of the center of the circle in local coordinates.

Implemented in Circles::Packing::HyperCircle, and Circles::Packing::EuclidCircle.

7.3.2.2 int Circle::index ( ) const

Get the index of the graph node associated with this Circle.

Returns

index of the graph node.

7.3.2.3 virtual QPointF Circles::Packing::Circle::projCenter ( ) const [pure virtual]

The center of the circle when projected into euclidean space (ie as it is on the monitor). For circles that exist in Euclidean space, this will be equal to its center()

Returns

the projected center of the circle.

Implemented in Circles::Packing::HyperCircle, and Circles::Packing::EuclidCircle.

7.3.2.4 virtual qreal Circles::Packing::Circle::projRadius ( ) const [pure virtual]

The radius of the circle when projected into euclidean space (ie as it is on the monitor). For circles that exist in Euclidean space, this will be equal to its radius().

Returns

the projected radius of the circle.

Implemented in Circles::Packing::HyperCircle, and Circles::Packing::EuclidCircle.

7.3.2.5 virtual qreal Circles::Packing::Circle::radius ( ) const [pure virtual]

The radius of the circle in its local space.

Returns

radius of the circle in its local space.

Implemented in Circles::Packing::HyperCircle, and Circles::Packing::EuclidCircle.

7.3.2.6 virtual bool Circles::Packing::Circle::setCenter (QPointF *c*) [pure virtual]

Attempt to set the center of the circle.

Parameters

c	the center to set

Returns

True if the center was set correctly. False otherwise.

Implemented in Circles::Packing::HyperCircle, and Circles::Packing::EuclidCircle.

7.3.2.7 void Circle::setIndex (int *index*)

Set the index of this circle

Parameters

index	the index of the associated node in the graph.

7.3.2.8 virtual void Circles::Packing::Circle::setRadius (qreal *r*) [pure virtual]

Set the radius of the circle to the specified value.

Parameters

r	The new radius of the circle. (in local space)

Implemented in Circles::Packing::HyperCircle, and Circles::Packing::EuclidCircle.

7.3.3 Friends And Related Function Documentation

7.3.3.1 bool operator< (const Circle & *lhs,* const Circle & *rhs*) [friend]

Order comparison for sorting in lists, etc.

Parameters

lhs	
rhs	

Returns

7.3.4 Member Data Documentation

7.3.4.1 QPointF Circles::Packing::Circle::_center [protected]

7.3.4.2 int Circles::Packing::Circle::_index [protected]

7.3.4.3 qreal Circles::Packing::Circle::_radius [protected]

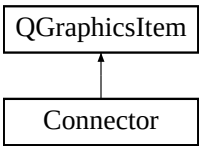
The documentation for this class was generated from the following files:

- packing/Circle.hpp
- packing/Circle.cpp

7.4 Connector Class Reference

`#include <Connector.hpp>`

Inheritance diagram for Connector:



Public Member Functions

- Connector (Node *n1, Node *n2)
- QRectF boundingRect () const Q_DECL_OVERRIDE
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget=nullptr) Q_DEC↩ L_OVERRIDE

7.4.1 Constructor & Destructor Documentation

7.4.1.1 Connector::Connector (Node * *n1*, Node * *n2*)

7.4.2 Member Function Documentation

7.4.2.1 QRectF Connector::boundingRect ( ) const

7.4.2.2 void Connector::paint (QPainter * *painter*, const QStyleOptionGraphicsItem * *option*, QWidget * *widget* = `nullptr`)

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

- graphics/Connector.hpp
- graphics/Connector.cpp

7.5 Circles::Graph::Edge Class Reference

`#include <Edge.hpp>`

Public Member Functions

- Edge ()
- Edge (int x, int y)
- Edge (const Edge &other)
- Edge & operator= (const Edge &other)
- int getX ()
- int getY ()
- void setX (int x)
- void setY (int y)
- void set (int x, int y)

Friends

- bool operator== (const Edge &lhs, const Edge &rhs)

7.5.1 Detailed Description

Defines an edge between two nodes in a graph. An edge is defined by the two nodes that it is co-incident to. The two defining nodes are constrained so that the lower-order one is the first argument.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 Circles::Graph::Edge::Edge ( )

Default constructor. Initializes the edge to (-1, -1)

7.5.2.2 Circles::Graph::Edge::Edge (int *x*, int *y*)

Construct an edge. The first node should be lower-order than the second.

Parameters

x	index of the first node of the edge
y	index of the second node of the edge.

7.5.2.3 Circles::Graph::Edge::Edge (const Edge & *other*)

7.5.3 Member Function Documentation

7.5.3.1 int Circles::Graph::Edge::getX ( )

7.5.3.2 int Circles::Graph::Edge::getY ( )

7.5.3.3 Edge & Circles::Graph::Edge::operator= (const Edge & *other*)

7.5.3.4 void Circles::Graph::Edge::set (int *x*, int *y*)

7.5.3.5 void Circles::Graph::Edge::setX (int *x*)

7.5.3.6 void Circles::Graph::Edge::setY (int *y*)

7.5.4 Friends And Related Function Documentation

7.5.4.1 bool operator== (const Edge & *lhs*, const Edge & *rhs*) [friend]

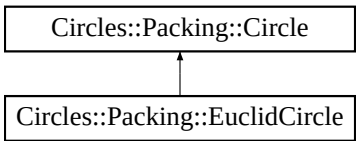
The documentation for this class was generated from the following files:

- graph/Edge.hpp
- graph/Edge.cpp

7.6 Circles::Packing::EuclidCircle Class Reference

`#include <EuclidCircle.hpp>`

Inheritance diagram for Circles::Packing::EuclidCircle:



Public Member Functions

- EuclidCircle ()
- EuclidCircle (const QPointF &center, qreal radius, int index)
- EuclidCircle (const EuclidCircle &other)
- EuclidCircle & operator= (const EuclidCircle &other)
- virtual QPointF center () const override
- virtual qreal radius () const override
- virtual QPointF projCenter () const override
- virtual qreal projRadius () const override
- virtual void setRadius (qreal r) override
- virtual bool setCenter (QPointF c) override

Friends

- bool operator== (const EuclidCircle &lhs, const EuclidCircle &rhs)

Additional Inherited Members

7.6.1 Detailed Description

A circle on the euclidean plane.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 Circles::Packing::EuclidCircle::EuclidCircle ( )

Construct an empty euclidean circle, centered at (0, 0) with radius 1.0, and index -1.

7.6.2.2 Circles::Packing::EuclidCircle::EuclidCircle (const QPointF & *center,* qreal *radius,* int *index*)

Construct a euclidean circle with a given center , radius, and index.

Parameters

center	Center point of the circle, in euclidean space.
radius	Radius of the circle.
index	Index of corresponding node in the underlying graph.

7.6.2.3 Circles::Packing::EuclidCircle::EuclidCircle (const EuclidCircle & *other*)

7.6.3 Member Function Documentation

7.6.3.1 QPointF Circles::Packing::EuclidCircle::center ( ) const [override], [virtual]

The center of the circle in it's local space.

Returns

The point of the center of the circle in local coordinates.

Implements Circles::Packing::Circle.

7.6.3.2 EuclidCircle & Circles::Packing::EuclidCircle::operator= (const EuclidCircle & *other*)

7.6.3.3 QPointF Circles::Packing::EuclidCircle::projCenter ( ) const [override], [virtual]

The center of the circle when projected into euclidean space (ie as it is on the monitor). For circles that exist in Euclidean space, this will be equal to its center()

Returns

the projected center of the circle.

Implements Circles::Packing::Circle.

7.6.3.4 qreal Circles::Packing::EuclidCircle::projRadius ( ) const [override], [virtual]

The radius of the circle when projected into euclidean space (ie as it is on the monitor). For circles that exist in Euclidean space, this will be equal to its radius().

Returns

the projected radius of the circle.

Implements Circles::Packing::Circle.

7.6.3.5 qreal Circles::Packing::EuclidCircle::radius ( ) const [override], [virtual]

The radius of the circle in its local space.

Returns

radius of the circle in its local space.

Implements Circles::Packing::Circle.

7.6.3.6 bool Circles::Packing::EuclidCircle::setCenter (QPointF *c*) [override], [virtual]

Attempt to set the center of the circle.

Parameters

c	the center to set

Returns

True if the center was set correctly. False otherwise.

Implements Circles::Packing::Circle.

7.6.3.7 void Circles::Packing::EuclidCircle::setRadius (qreal *r*) [override], [virtual]

Set the radius of the circle to the specified value.

Parameters

r	The new radius of the circle. (in local space)

Implements Circles::Packing::Circle.

7.6.4 Friends And Related Function Documentation

7.6.4.1 bool operator== (const EuclidCircle & *lhs*, const EuclidCircle & *rhs*) [friend]

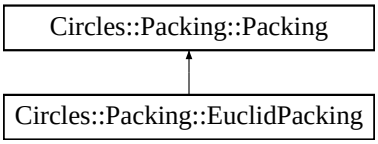
The documentation for this class was generated from the following files:

- packing/EuclidCircle.hpp
- packing/EuclidCircle.cpp

7.7 Circles::Packing::EuclidPacking Class Reference

`#include <EuclidPacking.hpp>`

Inheritance diagram for Circles::Packing::EuclidPacking:



Public Member Functions

- EuclidPacking ()
- EuclidPacking (std::shared_ptr< Graph::Graph > g)
- EuclidPacking (std::shared_ptr< Graph::Graph > g, const QList< Circle * > &circles)
- EuclidPacking (const EuclidPacking &other)
- EuclidPacking (EuclidPacking &&other)
- EuclidPacking & operator= (const EuclidPacking &other)
- EuclidPacking & operator= (EuclidPacking &&other)

Friends

- bool operator== (const EuclidPacking &lhs, const EuclidPacking &rhs)

Additional Inherited Members

7.7.1 Detailed Description

A packing on the euclidean plane. Holds EuclidCircles.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 EuclidPacking::EuclidPacking ( )

Construct an empty packing with no underlying graph. (don't use this)

7.7.2.2 Circles::Packing::EuclidPacking::EuclidPacking (std::shared_ptr< Graph::Graph > *g*)

Construct a new euclidean packing with a specified underlying graph. All circles will be initialized to radius 1 and center 0,0.

Parameters

g	pointer to underlying graph.

7.7.2.3 Circles::Packing::EuclidPacking::EuclidPacking (std::shared_ptr< Graph::Graph > *g*, const QList< Circle * > & *circles*)

Construct a new euclidean packing with a specified underlying graph and pre-defined circles.

Parameters

g	the underlying graph.
circles	List of circles to pre-setup. Dupliecate circles and circles which do not correspond to nodes present in the graph will be removed. Any nodes without circles in the list will have their circles initialized to radius 1.0 and center 0,0.

7.7.2.4 Circles::Packing::EuclidPacking::EuclidPacking (const EuclidPacking & *other*)

Copy reference to node as well as deep-copy of circles.

7.7.2.5 Circles::Packing::EuclidPacking::EuclidPacking (EuclidPacking && *other*)

7.7.3 Member Function Documentation

7.7.3.1 EuclidPacking& Circles::Packing::EuclidPacking::operator= (const EuclidPacking & *other*)

7.7.3.2 EuclidPacking& Circles::Packing::EuclidPacking::operator= (EuclidPacking && *other*)

7.7.4 Friends And Related Function Documentation

7.7.4.1 bool operator== (const EuclidPacking & *lhs*, const EuclidPacking & *rhs*) [friend]

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

- packing/EuclidPacking.hpp
- packing/EuclidPacking.cpp

7.8 Circles::Graph::Graph Class Reference

`#include <Graph.hpp>`

Public Member Functions

- Graph ()
- Graph (const Graph &other)
- Graph (Graph &&other)
- Graph & operator= (const Graph &other)
- Graph & operator= (Graph &&other)
- void addEdge (Node x, Node y)
- void addEdge (Edge e)
- void removeEdge (Node x, Node y)
- bool hasEdge (Node x, Node y) const

- bool hasFullFlower (Node n)
- bool isBoundary (Node n)
- std::unique_ptr< QList< Node > > getNodes () const
- std::unique_ptr< QList< Edge > > getEdges () const
- QList< Node > & neighbours (Node i) const
- QList< Node > & sortedNeighbours (Node n)
- QList< Node > boundary ()

7.8.1 Detailed Description

Implements a mathematical graph. The graph also stores information about the boundary nodes and edges.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Circles::Graph::Graph::Graph ( )

7.8.2.2 Circles::Graph::Graph::Graph (const Graph & *other*)

7.8.2.3 Circles::Graph::Graph::Graph (Graph && *other*)

7.8.3 Member Function Documentation

7.8.3.1 void Circles::Graph::Graph::addEdge (Node *x*, Node *y*)

Add an edge to the graph. Note that edges are symmetric. eg. AB == BA

Parameters

x	index of one end of the edge
y	indexc of the other end of the edge.

7.8.3.2 void Graph::addEdge (Edge *e*)

7.8.3.3 QList< Node > Graph::boundary ( )

Get a sorted list of nodes that make up the boundary of the graph.

Returns

sorted list of nodes.

7.8.3.4 std::unique_ptr< QList< Edge > > Circles::Graph::Graph::getEdges ( ) const

Get a set of all edges in the graph. The pairs returned will always have the lowest-order node index first.

Returns

A set of node pairs representing the edges of the graph.

7.8.3.5 std::unique_ptr< QList< Node > > Circles::Graph::Graph::getNodes ( ) const

Get the set of nodes that are present in the graph.

Returns

Set of the nodes present in the graph

7.8.3.6 bool Circles::Graph::Graph::hasEdge (Node *x*, Node *y*) const

Deterimine if the graph contains the edge between two nodes.

Parameters

x	the first node
y	the second node

Returns

True if the edge is contained in the graph, otherwise false.

7.8.3.7 bool Graph::hasFullFlower (Node *n*)

Determine if a specified node has a full flower, ie. if the neighbours of the node form a ring around it.

Parameters

n	node to query

Returns

True if the node has a full flower, otherwise False.

7.8.3.8 bool Graph::isBoundary (Node *n*)

Deterimine if a node is in the boundary of the graph.

Parameters

n	The node to query

Returns

True if the node is in the boundary, otherwise false.

7.8.3.9 QList< Node > & Circles::Graph::Graph::neighbours (Node *i*) const

Get the set of nodes which are adjacent to the specified node. Does not require sorting of the nodes lst.

Parameters

i	Node to query

Returns

Set of nodes which are ajacent to the specified node.

7.8.3.10 Graph & Circles::Graph::Graph::operator= (const Graph & *other*)

7.8.3.11 Graph & Circles::Graph::Graph::operator= (Graph && *other*)

7.8.3.12 void Circles::Graph::Graph::removeEdge (Node *x*, Node *y*)

Removes an edge from the graph, if it exists.

Parameters

x	index of first node which defines edge
y	index of second node which defines edge

7.8.3.13 QList< Node > Graph::sortedNeighbours (Node *n*)

Attempt to get a list of the ajacent nodes of a specified node such that two nodes ajacent in the list are also ajacent in the graph.

Parameters

i	

Returns

The sorted list, if it exists. Otherwise return an empty list.

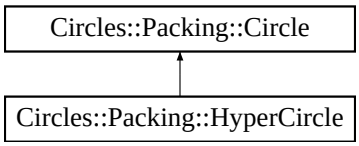
The documentation for this class was generated from the following files:

- graph/Graph.hpp
- graph/Graph.cpp

7.9 Circles::Packing::HyperCircle Class Reference

`#include <HyperCircle.hpp>`

Inheritance diagram for Circles::Packing::HyperCircle:



Public Member Functions

- HyperCircle ()
- HyperCircle (const QPointF &center, qreal radius, int index)
- HyperCircle (const HyperCircle &other)
- HyperCircle & operator= (const HyperCircle &other)
- virtual QPointF center () const override
- virtual qreal radius () const override
- virtual QPointF projCenter () const override
- virtual qreal projRadius () const override
- virtual void setRadius (qreal r) override
- virtual bool setCenter (QPointF c) override

Friends

- bool operator== (const HyperCircle &lhs, const HyperCircle &rhs)

Additional Inherited Members

7.9.1 Detailed Description

A circle in the hyperbolic poincare disc.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 Circles::Packing::HyperCircle::HyperCircle ( )

Construct an empty euclidean circle, centered at (0, 0) with radius 1.0, and index -1.

7.9.2.2 Circles::Packing::HyperCircle::HyperCircle (const QPointF & *center,* qreal *radius,* int *index*)

Construct a Hyperbolic circle with a given center , radius, and index.

Parameters

center	Center point of the circle, in hyperbolic disc space.
radius	Hyperbolic Radius of the circle.
index	Index of corresponding node in the underlying graph.

7.9.2.3 Circles::Packing::HyperCircle::HyperCircle (const HyperCircle & *other*)

7.9.3 Member Function Documentation

7.9.3.1 QPointF Circles::Packing::HyperCircle::center ( ) const [override], [virtual]

The center of the circle in it's local space.

Returns

The point of the center of the circle in local coordinates.

Implements Circles::Packing::Circle.

7.9.3.2 HyperCircle & Circles::Packing::HyperCircle::operator= (const HyperCircle & *other*)

7.9.3.3 QPointF Circles::Packing::HyperCircle::projCenter ( ) const [override], [virtual]

The center of the circle when projected into euclidean space (ie as it is on the monitor). For circles that exist in Euclidean space, this will be equal to its center()

Returns

the projected center of the circle.

Implements Circles::Packing::Circle.

7.9.3.4 qreal Circles::Packing::HyperCircle::projRadius ( ) const [override], [virtual]

The radius of the circle when projected into euclidean space (ie as it is on the monitor). For circles that exist in Euclidean space, this will be equal to its radius().

Returns

the projected radius of the circle.

Implements Circles::Packing::Circle.

7.9.3.5 qreal Circles::Packing::HyperCircle::radius ( ) const [override], [virtual]

The radius of the circle in its local space.

Returns

radius of the circle in its local space.

Implements Circles::Packing::Circle.

7.9.3.6 bool Circles::Packing::HyperCircle::setCenter (QPointF *c*) [override], [virtual]

Attempt to set the center of the circle.

Parameters

c	the center to set

Returns

True if the center was set correctly. False otherwise.

Implements Circles::Packing::Circle.

7.9.3.7 void Circles::Packing::HyperCircle::setRadius (qreal *r*) [override], [virtual]

Set the radius of the circle to the specified value.

Parameters

r	The new radius of the circle. (in local space)

Implements Circles::Packing::Circle.

7.9.4 Friends And Related Function Documentation

7.9.4.1 bool operator== (const HyperCircle & *lhs*, const HyperCircle & *rhs*) [friend]

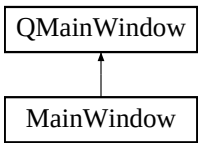
The documentation for this class was generated from the following files:

- packing/HyperCircle.hpp
- packing/HyperCircle.cpp

7.10 MainWindow Class Reference

`#include <MainWindow.hpp>`

Inheritance diagram for MainWindow:



Public Member Functions

- MainWindow (QWidget *parent=0)
- ~MainWindow ()

7.10.1 Constructor & Destructor Documentation

7.10.1.1 MainWindow::MainWindow (QWidget * *parent* = 0) [explicit]

7.10.1.2 MainWindow::~MainWindow ( )

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

- ui/MainWindow.hpp
- ui/MainWindow.cpp

7.11 Node Class Reference

The Node class represents the metadata of a circle in a circle packing.

`#include <Node.hpp>`

Public Member Functions

- Node (const Node *n)
- Node (int id)
- Node (int id, const QPointF &position, qreal radius=0)
- ~Node ()
- int getId ()
- void setId (int value)
- qreal getRadius ()
- void setRadius (const qreal value)
- QColor getColor ()
- void setColor (const QColor &value)
- void addNeibhour (Node *node)
- void delNeibhour (Node *node)
- void purgeNeibhours ()
- bool isNeibhour (Node *node)
- void sortNeibhours ()
- QList< Node * > getNeibhours ()

- int getNeibhourCount ()
- void setPosition (const QPointF &position)
- QPointF getPosition ()
- bool hasPosition ()
- void delPosition ()
- bool hasFullFlower ()

Static Public Member Functions

- static QList< Node * > generateHexArray (const QRectF &area, qreal radius)
- static QList< Node * > generateHexArray (const QPointF &startpos, int w, int h, qreal radius)

Protected Attributes

- int id
- QPointF position
- qreal radius
- QList< Node * > neibhours
- QColor color
- bool bHasPosition =false
- bool sortedNeibhours =false

7.11.1 Detailed Description

The Node class represents the metadata of a circle in a circle packing.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 Node::Node (const Node * *n*)

Copy constructor. Note that neibhour relationships are not copied.

Parameters

n	Node to copy

7.11.2.2 Node::Node (int *id*)

Construct a Node without position or radius.

Parameters

id	unique id of the node

7.11.2.3 Node::Node (int *id*, const QPointF & *position*, qreal *radius* = 0)

Construct a node with a given position and radius.

Parameters

id	Unique id of the node
position	Position of the node
radius	radius of the node

7.11.2.4 Node::∼Node ( )

7.11.3 Member Function Documentation

7.11.3.1 void Node::addNeibhour (Node * *node*)

7.11.3.2 void Node::delNeibhour (Node * *node*)

7.11.3.3 void Node::delPosition ( )

7.11.3.4 QList< Node * > Node::generateHexArray (const QRectF & *area,* qreal *radius*) [static]

Generates a hex tiling of circles with specified radius such that circles are guaranteed to cover at least the specified area

Parameters

area	the area to cover
radius	radius of the circles

Returns

list of nodes representing the circles created.

7.11.3.5 QList< Node * > Node::generateHexArray (const QPointF & *startpos,* int *w,* int *h,* qreal *radius*) [static]

Parameters

startpos	the location of the top-left circle
w	width, in circles, of the circle array
h	height, in circles, of the circle array. Should be odd.
radius	the radius of each circle.

Returns

list of nodes representing the circles created.

7.11.3.6 QColor Node::getColor ( )

7.11.3.7 int Node::getId ( )

7.11.3.8 int Node::getNeibhourCount ( )

7.11.3.9 QList< Node * > Node::getNeibhours ( )

7.11.3.10 QPointF Node::getPosition ( )

7.11.3.11 qreal Node::getRadius ( )

7.11.3.12 bool Node::hasFullFlower ( )

7.11.3.13 bool Node::hasPosition ( )

7.11.3.14 bool Node::isNeibhour (Node * *node*)

7.11.3.15 void Node::purgeNeibhours ( )

7.11.3.16 void Node::setColor (const QColor & *value*)

7.11.3.17 void Node::setId (int *value*)

7.11.3.18 void Node::setPosition (const QPointF & *position*)

7.11.3.19 void Node::setRadius (const qreal *value*)

7.11.3.20 void Node::sortNeibhours ( )

7.11.4 Member Data Documentation

7.11.4.1 bool Node::bHasPosition =false [protected]

7.11.4.2 QColor Node::color [protected]

7.11.4.3 int Node::id [protected]

7.11.4.4 QList<Node*> Node::neibhours [protected]

7.11.4.5 QPointF Node::position [protected]

7.11.4.6 qreal Node::radius [protected]

7.11.4.7 bool Node::sortedNeibhours =false [protected]

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

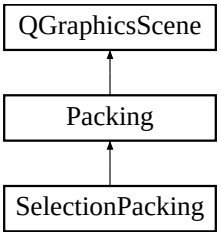
- Node.hpp
- Node.cpp

7.12 Packing Class Reference

The Packing class is an abstract class which defines a circle packing. A circle packing may be either euclidean or hyperbolic, as found in EuclideanPacking and HyperbolicPacking.

`#include <Packing.hpp>`

Inheritance diagram for Packing:



Public Slots

- void setDrawCenters (bool d)
- void setDrawLinks (bool d)
- void setDrawCircles (bool d)
- void setDrawIndicies (bool d)
- void setDrawBoundary (bool d)
- void repack (qreal epsilon, qreal outerRadius)

  Compute the radii of the circles that will result in an actual packing.
- void layout (int centerCircle)

  lay-out circles once radii have been computed.

Signals

- void newNodeSelected (Node ∗n)

Public Member Functions

- Packing (const Packing ∗p)
- Packing (PackingType type=PackingType::EuclideanPacking)

  Create a new, empty Packing object with specified geometry.
- Packing (QList< Node ∗ > nodes, PackingType type=PackingType::EuclideanPacking)

  Generate a packing containing the specified nodes.
- ∼Packing ()
- void setPackingType (PackingType type)

  setPackingType sets the geometry of the packing
- PackingType getType ()

  getType
- bool getDrawCenters ()
- bool getDrawLinks ()
- bool getDrawCircles ()
- bool getDrawIndicies ()
- void addNode (Node ∗n)

  addNode Adds a node to the packing.
- void addNode_fast (Node ∗n)

  Add a node without re-computing connectors. You must manually call recompute_connectors() after adding all nodes.
- void delNode (Node ∗n)
- void delNode_fast (Node ∗n)
- virtual void mousePressEvent (QGraphicsSceneMouseEvent ∗mouseEvent) Q_DECL_OVERRIDE
- void recomputeConnectors ()

  recomputeConnectors re-computes the coordinates of the connectors between circle-centers.
- QList< Node ∗ > getNodes ()
- bool isInterior (Node ∗n)

  Determine if given node is interior to packing.
- bool isExterior (Node ∗n)
- void refreshCircles ()
- void resetIds ()

Public Attributes

- int centerCircleID = -1

Protected Member Functions

- void layout_hyperbolic (int centerCircle)
- void layout_euclidean (int centerCircle)
- qreal angle (Node *r, Node *ra, Node *rb)

  Compute the angle formed by the tangent circles of 3 nodes.
- qreal angle_euclidean (Node *r, Node *ra, Node *rb)
- qreal angle_hyperbolic (Node *r, Node *ra, Node *rb)
- qreal anglesum (Node *r)

  returns sum of all angles formed with ajacent nodes
- void addCircle (Node *n)
- void purgeCircles ()
- void drawForeground (QPainter *painter, const QRectF &rect) Q_DECL_OVERRIDE

Protected Attributes

- bool drawCenters =false
- bool drawLinks =false
- bool drawCircles =true
- bool drawIndicies =false
- PackingType type
- QList< Node * > nodes
- QList< Node * > boundaryNodes
- QList< Circle * > circles
- QList< Connector * > connectors
- Circle * selectedCircle = nullptr

7.12.1 Detailed Description

The Packing class is an abstract class which defines a circle packing. A circle packing may be either euclidean or hyperbolic, as found in EuclideanPacking and HyperbolicPacking.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 Packing::Packing (const Packing * *p*)

Copy constructor. Nodes are deep-copied.

Parameters

p	pointer to packing to be copied.

7.12.2.2 Packing::Packing (PackingType *type* = PackingType::EuclideanPacking)

Create a new, empty Packing object with specified geometry.

Parameters

type	either PackingType::EuclideanPacking or PackingType::HyperbolicPacking

7.12.2.3 Packing::Packing (QList< Node * > *nodes*, PackingType *type* = PackingType::EuclideanPacking)

Generate a packing containing the specified nodes.

Parameters

nodes	
type	

7.12.2.4 Packing::~Packing ( )

7.12.3 Member Function Documentation

7.12.3.1 void Packing::addCircle (Node * *n*) [protected]

7.12.3.2 void Packing::addNode (Node * *n*)

addNode Adds a node to the packing.

Parameters

n	node to add.

7.12.3.3 void Packing::addNode_fast (Node * *n*)

Add a node without re-computing connectors. You must manually call recompute_connectors() after adding all nodes.

Parameters

n	node to add

7.12.3.4 qreal Packing::angle (Node * *r*, Node * *ra*, Node * *rb*) [protected]

Compute the angle formed by the tangent circles of 3 nodes.

Parameters

r	the center node of the angle
ra	one leg of the angle
rb	one leg of the angle

Returns

the angle formed, in radians.

7.12.3.5 qreal Packing::angle_euclidean (Node * *r*, Node * *ra*, Node * *rb*) [protected]

7.12.3.6 qreal Packing::angle_hyperbolic (Node * *r*, Node * *ra*, Node * *rb*) [protected]

7.12.3.7 qreal Packing::anglesum (Node * *r*) [protected]

returns sum of all angles formed with ajacent nodes

Parameters

r	node

Returns

angle sum

7.12.3.8 void Packing::delNode (Node * *n*)

7.12.3.9 void Packing::delNode_fast (Node * *n*)

MUST CALL REFRESHCIRCLES AFTER RUNNING

Parameters

n	

7.12.3.10 void Packing::drawForeground (QPainter * *painter,* const QRectF & *rect*) [protected]

7.12.3.11 bool Packing::getDrawCenters ( )

7.12.3.12 bool Packing::getDrawCircles ( )

7.12.3.13 bool Packing::getDrawIndicies ( )

7.12.3.14 bool Packing::getDrawLinks ( )

7.12.3.15 QList< Node * > Packing::getNodes ( )

7.12.3.16 PackingType Packing::getType ( )

getType

Returns

geometry of the packing.

7.12.3.17 bool Packing::isExterior (Node * *n*)

7.12.3.18 bool Packing::isInterior (Node * *n*)

Determine if given node is interior to packing.

Parameters

n	node

Returns

true if node is interior to packing. Otherwise False.

7.12.3.19 void Packing::layout (int *centerCircle*) [slot]

lay-out circles once radii have been computed.

Parameters

centerCircle	index of circle to place at center of plane/disc

7.12.3.20 void Packing::layout_euclidean (int *centerCircle*) [protected]

7.12.3.21 void Packing::layout_hyperbolic (int *centerCircle*) [protected]

7.12.3.22 void Packing::mousePressEvent (QGraphicsSceneMouseEvent * *mouseEvent*) [virtual]

Reimplemented in SelectionPacking.

7.12.3.23 void Packing::newNodeSelected (Node * *n*) [signal]

7.12.3.24 void Packing::purgeCircles ( ) [protected]

7.12.3.25 void Packing::recomputeConnectors ( )

recomputeConnectors re-computes the coordinates of the connectors between circle-centers.

7.12.3.26 void Packing::refreshCircles ( )

Re-creates all circles in teh packing

7.12.3.27 void Packing::repack (qreal *epsilon,* qreal *outerRadius*) [slot]

Compute the radii of the circles that will result in an actual packing.

Parameters

epsilon	epsilon value to determine completeness. Small.
outerRadius	Radius of boundary circles. Large.

7.12.3.28 void Packing::resetIds ( )

Reset the ids of all nodes so that they lie in the range [0, n].

7.12.3.29 void Packing::setDrawBoundary (bool *d*) [slot]

7.12.3.30 void Packing::setDrawCenters (bool *d*) [slot]

7.12.3.31 void Packing::setDrawCircles (bool *d*) [slot]

7.12.3.32 void Packing::setDrawIndicies (bool *d*) [slot]

7.12.3.33 void Packing::setDrawLinks (bool *d*) [slot]

7.12.3.34 void Packing::setPackingType (PackingType *type*)

setPackingType sets the geometry of the packing

Parameters

type	either PackingType::EuclideanPacking or PackingType::HyperbolicPacking

7.12.4 Member Data Documentation

7.12.4.1 QList<Node*> Packing::boundaryNodes [protected]

7.12.4.2 int Packing::centerCircleID = -1

7.12.4.3 QList<Circle*> Packing::circles [protected]

7.12.4.4 QList<Connector*> Packing::connectors [protected]

7.12.4.5 bool Packing::drawCenters =false [protected]

7.12.4.6 bool Packing::drawCircles =true [protected]

7.12.4.7 bool Packing::drawIndicies =false [protected]

7.12.4.8 bool Packing::drawLinks =false [protected]

7.12.4.9 QList<Node*> Packing::nodes [protected]

7.12.4.10 Circle* Packing::selectedCircle = nullptr [protected]

7.12.4.11 PackingType Packing::type [protected]

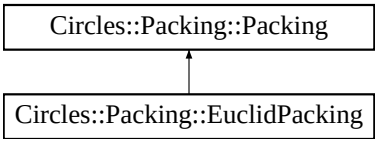
The documentation for this class was generated from the following files:

- graphics/Packing.hpp
- graphics/Packing.cpp

7.13 Circles::Packing::Packing Class Reference

`#include <Packing.hpp>`

Inheritance diagram for Circles::Packing::Packing:



Public Member Functions

- virtual void repack (qreal epsilon, qreal outerRadius)=0
- virtual void layout (int centerNode)=0
- virtual qreal angle (const QPointF &p, const QPointF &p1, const QPointF &p2) const =0
- const Graph::Graph & graph () const
- QMap< int, Circle * > circles () const
- const Circle * circle (int index) const

Protected Member Functions

- qreal anglesum (const Circle &c) const

Protected Attributes

- std::shared_ptr< Graph::Graph > _graph
- QMap< int, std::unique_ptr< Circle > > _circles

7.13.1 Detailed Description

Abstract base class of the EuclidPacking and the HyperPacking, which represent a circle packing over a specific Graph.

Circle packings consist of a number of circles which lie tangent to eachother.

7.13.2 Member Function Documentation

7.13.2.1 virtual qreal Circles::Packing::Packing::angle (const QPointF & *p*, const QPointF & *p1*, const QPointF & *p2*) const [pure virtual]

Compute the angle <p,p1,p2 in the local space of the packing.

Parameters

point	which represents the vertex point.
point	which defines one ray of the angle.
point	which defines the other ray of the angle.

Returns

The angle formed by the points, in radians in range [0, pi]

7.13.2.2 qreal Packing::anglesum (const Circle & *c*) const [protected]

Compute the sum of the angles formed by a circle and its neighbours.

Parameters

c	the circle to consider.

Returns

sum of angles in radians, in range [0, 2pi]

7.13.2.3 const Circle * Packing::circle (int *index*) const

Get a circle based on its index.

Parameters

index	index of the circle

Returns

const reference to circle with specified index.

7.13.2.4 QMap< int, Circle * > Packing::circles ( ) const

Get a list of all circles in the packing, addressable by their indicies.

Returns

Qhash of index -> circle elements for the packing.

7.13.2.5 const Graph::Graph & Packing::graph ( ) const

Get a reference to the underlying graph of the packing.

Returns

const reference to graph object underlying the packing.

7.13.2.6 virtual void Circles::Packing::Packing::layout (int *centerNode*) [pure virtual]

Lay out the circles such that neighbouring circles are tangent to one another.

Parameters

centerNode	index of the circle to place at the center of the packing.

7.13.2.7 virtual void Circles::Packing::Packing::repack (qreal *epsilon*, qreal *outerRadius*) [pure virtual]

Set boundary circles to be of a specified radius and then modify other circles to form a proper packing.

Parameters

epsilon	tolerance value when determining angle sum
outerRadius	Radius of the boundary circles.

7.13.3 Member Data Documentation

7.13.3.1 QMap<int, std::unique_ptr<Circle> > Circles::Packing::Packing::_circles [protected]

7.13.3.2 std::shared_ptr<Graph::Graph> Circles::Packing::Packing::_graph [protected]

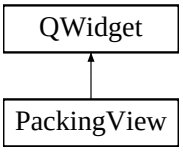
The documentation for this class was generated from the following files:

- packing/Packing.hpp
- packing/Packing.cpp

7.14 PackingView Class Reference

`#include <PackingView.hpp>`

Inheritance diagram for PackingView:



Public Slots

- void setPacking (Packing *p)

Public Member Functions

- PackingView (QWidget *parent=0)
- PackingView (Packing *p, QWidget *parent=0)
- ~PackingView ()

7.14.1 Constructor & Destructor Documentation

7.14.1.1 PackingView::PackingView (QWidget * *parent* = `0`) `[explicit]`

7.14.1.2 PackingView::PackingView (Packing * *p*, QWidget * *parent* = `0`)

7.14.1.3 PackingView::~PackingView ( )

7.14.2 Member Function Documentation

7.14.2.1 void PackingView::setPacking (Packing * *p*) `[slot]`

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

- ui/PackingView.hpp
- ui/PackingView.cpp

7.15 PFile Class Reference

`#include <PFile.hpp>`

Public Member Functions

- PFile (QString filename)
- Packing * generatePacking ()

7.15.1 Constructor & Destructor Documentation

7.15.1.1 PFile::PFile (QString *filename*)

7.15.2 Member Function Documentation

7.15.2.1 Packing * PFile::generatePacking ( )

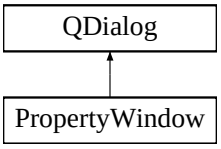
The documentation for this class was generated from the following files:

- PFile.hpp
- PFile.cpp

7.16 PropertyWindow Class Reference

`#include <PropertyWindow.hpp>`

Inheritance diagram for PropertyWindow:



Public Slots

- void setNode (Node *n)
- void refresh ()

Public Member Functions

- PropertyWindow (QWidget *parent=0)
- ~PropertyWindow ()

7.16.1 Constructor & Destructor Documentation

7.16.1.1 PropertyWindow::PropertyWindow (QWidget * *parent* = `0`) `[explicit]`

7.16.1.2 PropertyWindow::~PropertyWindow ( )

7.16.2 Member Function Documentation

7.16.2.1 void PropertyWindow::refresh ( ) `[slot]`

Updates the visual display using information from the currently selected node.

7.16.2.2 void PropertyWindow::setNode (Node * *n*) `[slot]`

Sets the node which is displayed in the property window.

Parameters

n	Node to display.

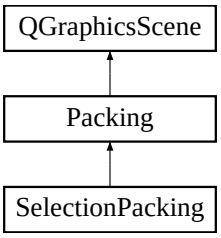
The documentation for this class was generated from the following files:

- ui/PropertyWindow.hpp
- ui/PropertyWindow.cpp

7.17 SelectionPacking Class Reference

`#include <SelectionPacking.hpp>`

Inheritance diagram for SelectionPacking:



Public Types

- enum MouseMode { MouseMode::None, MouseMode::Add, MouseMode::Subtract }

Public Member Functions

- SelectionPacking (PackingType type=PackingType::EuclideanPacking)
- SelectionPacking (QList< Node * > nodes, PackingType type=PackingType::HyperbolicPacking)
- bool addToSelection (Circle *c)

  Attempts to add a circle c to the collection of selected circles If a previous circle is in the selection and the new circle is not ajacent to that circle, then the circle will not be added.
- void removeFromSelection (Circle *c)
- bool isInSelection (Circle *c)
- void clearSelection ()
- void mousePressEvent (QGraphicsSceneMouseEvent *mouseEvent) Q_DECL_OVERRIDE
- void mouseMoveEvent (QGraphicsSceneMouseEvent *event) Q_DECL_OVERRIDE
- void mouseReleaseEvent (QGraphicsSceneMouseEvent *event) Q_DECL_OVERRIDE

Additional Inherited Members

7.17.1 Member Enumeration Documentation

7.17.1.1 enum SelectionPacking::MouseMode [strong]

Enumerator

None

Add

Subtract

7.17.2 Constructor & Destructor Documentation

7.17.2.1 SelectionPacking::SelectionPacking (PackingType *type* = PackingType::EuclideanPacking)

7.17.2.2 SelectionPacking::SelectionPacking (QList< Node * > *nodes*, PackingType *type* = PackingType::HyperbolicPacking)

7.17.3 Member Function Documentation

7.17.3.1 bool SelectionPacking::addToSelection (Circle * *c*)

Attempts to add a circle c to the collection of selected circles If a previous circle is in the selection and the new circle is not ajacent to that circle, then the circle will not be added.

Parameters

c	

Returns

7.17.3.2 void SelectionPacking::clearSelection ( )

7.17.3.3 bool SelectionPacking::isInSelection (Circle * *c*)

7.17.3.4 void SelectionPacking::mouseMoveEvent (QGraphicsSceneMouseEvent * *event*)

7.17.3.5 void SelectionPacking::mousePressEvent (QGraphicsSceneMouseEvent * *mouseEvent*) [virtual]

Reimplemented from Packing.

7.17.3.6 void SelectionPacking::mouseReleaseEvent (QGraphicsSceneMouseEvent * *event*)

7.17.3.7 void SelectionPacking::removeFromSelection (Circle * *c*)

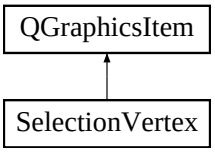
The documentation for this class was generated from the following files:

- graphics/SelectionPacking.hpp
- graphics/SelectionPacking.cpp

7.18 SelectionVertex Class Reference

`#include <SelectionVertex.hpp>`

Inheritance diagram for SelectionVertex:



Public Member Functions

- SelectionVertex (QPointF position, QColor color=QColor(255, 0, 0))
- QRectF boundingRect () const Q_DECL_OVERRIDE
- void paint (QPainter ∗painter, const QStyleOptionGraphicsItem ∗option, QWidget ∗widget) Q_DECL_OVE↩ RRIDE

Static Public Attributes

- static const qreal size = 15
- static const qreal thickness = 3

7.18.1 Detailed Description

Class which represents a selected vertex in the shape selector view.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 SelectionVertex::SelectionVertex (QPointF *position,* QColor *color* = `QColor(255, 0, 0)`)

7.18.3 Member Function Documentation

7.18.3.1 QRectF SelectionVertex::boundingRect ( ) const

7.18.3.2 void SelectionVertex::paint (QPainter ∗ *painter,* const QStyleOptionGraphicsItem ∗ *option,* QWidget ∗ *widget*)

7.18.4 Member Data Documentation

7.18.4.1 const qreal SelectionVertex::size = 15 `[static]`

7.18.4.2 const qreal SelectionVertex::thickness = 3 `[static]`

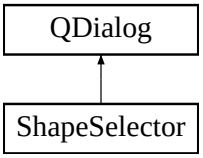
The documentation for this class was generated from the following files:

- graphics/SelectionVertex.hpp
- graphics/SelectionVertex.cpp

7.19 ShapeSelector Class Reference

`#include <ShapeSelector.hpp>`

Inheritance diagram for ShapeSelector:



Signals

- void packingAccepted (Packing ∗p)

Public Member Functions

- ShapeSelector (QWidget ∗parent=0)
- ∼ShapeSelector ()

7.19.1 Constructor & Destructor Documentation

7.19.1.1 ShapeSelector::ShapeSelector (QWidget ∗ *parent* = 0) [explicit]

7.19.1.2 ShapeSelector::∼ShapeSelector ( )

7.19.2 Member Function Documentation

7.19.2.1 void ShapeSelector::packingAccepted (Packing ∗ *p*) [signal]

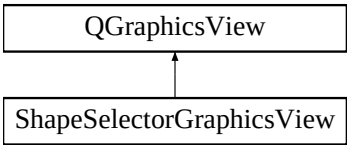
The documentation for this class was generated from the following files:

- ui/ShapeSelector.hpp
- ui/ShapeSelector.cpp

7.20 ShapeSelectorGraphicsView Class Reference

`#include <ShapeSelectorGraphicsView.hpp>`

Inheritance diagram for ShapeSelectorGraphicsView:



Signals

- void gotClick (QPointF scenepos)

Public Member Functions

- ShapeSelectorGraphicsView (QWidget ∗parent=nullptr)
- ShapeSelectorGraphicsView (QGraphicsScene ∗scene, QWidget ∗parent=nullptr)
- virtual int heightForWidth (int w) const
- virtual bool hasHeightForWidth () const

Protected Member Functions

- virtual void resizeEvent (QResizeEvent ∗event)
- virtual void mousePressEvent (QMouseEvent ∗event)

7.20.1 Constructor & Destructor Documentation

7.20.1.1 ShapeSelectorGraphicsView::ShapeSelectorGraphicsView (QWidget * *parent* = `nullptr`)

7.20.1.2 ShapeSelectorGraphicsView::ShapeSelectorGraphicsView (QGraphicsScene * *scene*, QWidget * *parent* = `nullptr`)

7.20.2 Member Function Documentation

7.20.2.1 void ShapeSelectorGraphicsView::gotClick (QPointF *scenepos*) `[signal]`

7.20.2.2 virtual bool ShapeSelectorGraphicsView::hasHeightForWidth ( ) const `[inline]`, `[virtual]`

7.20.2.3 virtual int ShapeSelectorGraphicsView::heightForWidth (int *w*) const `[inline]`, `[virtual]`

7.20.2.4 void ShapeSelectorGraphicsView::mousePressEvent (QMouseEvent * *event*) `[protected]`, `[virtual]`

7.20.2.5 void ShapeSelectorGraphicsView::resizeEvent (QResizeEvent * *event*) `[protected]`, `[virtual]`

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

- graphics/ShapeSelectorGraphicsView.hpp
- graphics/ShapeSelectorGraphicsView.cpp

Chapter 8

File Documentation

8.1 graph/Edge.cpp File Reference

```
#include "Edge.hpp"
```

8.2 graph/Edge.hpp File Reference

Classes

- class Circles::Graph::Edge

Namespaces

- Circles
- Circles::Graph

Functions

- bool Circles::Graph::operator== (const Edge &lhs, const Edge &rhs)

8.3 graph/Graph.cpp File Reference

```
#include "Graph.hpp"
```

8.4 graph/Graph.hpp File Reference

```
#include <QHash>
#include <QSet>
#include <QList>
#include <memory>
#include <graph/Edge.hpp>
```

Classes

- class Circles::Graph::Graph

Namespaces

- Circles
- Circles::Graph

Typedefs

- typedef int Circles::Graph::Node

8.5 graphics/Boundary.cpp File Reference

```
#include "Boundary.hpp"
#include "QtWidgets"
#include <cmath>
```

Macros

- #define PI 3.141592654

8.5.1 Macro Definition Documentation

8.5.1.1 #define PI 3.141592654

8.6 graphics/Boundary.hpp File Reference

```
#include <QWidget>
#include <QGraphicsItem>
```

Classes

- class Boundary

8.7 graphics/Circle.cpp File Reference

```
#include "Circle.hpp"
#include <cmath>
#include <QtWidgets>
#include <QPainterPath>
#include "Packing.hpp"
```

8.8 packing/Circle.cpp File Reference

```
#include "Circle.hpp"
```

8.9 graphics/Circle.hpp File Reference

```
#include <QGraphicsItem>
#include <QWidget>
#include <QPainterPath>
#include "../Node.hpp"
#include "Packing.hpp"
```

Classes

- class Circle

  The Circle class represents an abstract circle. A circle may exist in either a hyperbolic or euclidean geometry.

8.10 packing/Circle.hpp File Reference

```
#include <QPointF>
```

Classes

- class Circles::Packing::Circle

Namespaces

- Circles
- Circles::Packing

Functions

- bool Circles::Packing::operator< (const Circle &lhs, const Circle &rhs)

8.11 graphics/Connector.cpp File Reference

```
#include "Connector.hpp"
#include <cmath>
#include <QtWidgets>
```

8.12 graphics/Connector.hpp File Reference

```
#include <QWidget>
#include <QGraphicsItem>
#include "../Node.hpp"
```

Classes

- class Connector

8.13 graphics/Packing.cpp File Reference

```
#include "Packing.hpp"
#include <cmath>
#include <complex>
#include <QWidget>
#include <QDebug>
#include "Boundary.hpp"
```

Macros

- #define PI 3.14159265358979323846264 33

8.13.1 Macro Definition Documentation

8.13.1.1 #define PI 3.14159265358979323846264 33

8.14 packing/Packing.cpp File Reference

```
#include <algorithm>
#include "Packing.hpp"
```

8.15 graphics/Packing.hpp File Reference

```
#include <QGraphicsScene>
#include <QGraphicsSceneMouseEvent>
#include <QWidget>
#include <QObject>
#include "../Node.hpp"
#include "Circle.hpp"
#include "Connector.hpp"
#include "Boundary.hpp"
```

Classes

- class Packing

The Packing class is an abstract class which defines a circle packing. A circle packing may be either euclidean or hyperbolic, as found in EuclideanPacking and HyperbolicPacking.

Enumerations

- enum PackingType { PackingType::EuclideanPacking, PackingType::HyperbolicPacking }

  The PackingType enum defines the geometries available to a Packing. A packing may either be euclidean, on the cartesian plane, or hyperbolic, on the poincare disc.

8.15.1 Enumeration Type Documentation

8.15.1.1 enum PackingType [strong]

The PackingType enum defines the geometries available to a Packing. A packing may either be euclidean, on the cartesian plane, or hyperbolic, on the poincare disc.

Enumerator

EuclideanPacking

HyperbolicPacking

8.16 packing/Packing.hpp File Reference

```
#include <memory>
#include <QList>
#include <QMap>
#include <graph/Graph.hpp>
#include <packing/Circle.hpp>
```

Classes

- class Circles::Packing::Packing

Namespaces

- Circles
- Circles::Packing

8.17 graphics/SelectionPacking.cpp File Reference

```
#include "SelectionPacking.hpp"
#include <typeinfo>
#include <QDebug>
```

8.18 graphics/SelectionPacking.hpp File Reference

```
#include <QWidget>
#include <QGraphicsScene>
#include <QGraphicsSceneMouseEvent>
#include "Packing.hpp"
#include "Circle.hpp"
```

Classes

- class SelectionPacking

8.19 graphics/SelectionVertex.cpp File Reference

```
#include "SelectionVertex.hpp"
#include <QPainter>
#include <QDebug>
#include <QStyleOptionGraphicsItem>
```

8.20 graphics/SelectionVertex.hpp File Reference

```
#include <QObject>
#include <QWidget>
#include <QGraphicsItem>
```

Classes

- class SelectionVertex

8.21 graphics/ShapeSelectorGraphicsView.cpp File Reference

```
#include "ShapeSelectorGraphicsView.hpp"
#include <QResizeEvent>
```

8.22 graphics/ShapeSelectorGraphicsView.hpp File Reference

```
#include <QObject>
#include <QWidget>
#include <QGraphicsScene>
#include <QGraphicsView>
#include <QPointF>
```

Classes

- class ShapeSelectorGraphicsView

8.23 main.cpp File Reference

```
#include "ui/MainWindow.hpp"
#include <QApplication>
```

Functions

- int main (int argc, char ∗argv[ ])

8.23.1 Function Documentation

8.23.1.1 int main (int *argc*, char ∗ *argv[ ]*)

8.24 Node.cpp File Reference

```
#include "Node.hpp"
#include <cmath>
```

Macros

- #define PI 3.14159265358979323846264 33

8.24.1 Macro Definition Documentation

8.24.1.1 #define PI 3.14159265358979323846264 33

8.25 Node.hpp File Reference

```
#include <QWidget>
```

Classes

- class Node

  The Node class represents the metadata of a circle in a circle packing.

8.26 packing/EuclidCircle.cpp File Reference

```
#include "EuclidCircle.hpp"
```

8.27 packing/EuclidCircle.hpp File Reference

```
#include <packing/Circle.hpp>
```

Classes

- class Circles::Packing::EuclidCircle

Namespaces

- Circles
- Circles::Packing

Functions

- bool Circles::Packing::operator== (const EuclidCircle &lhs, const EuclidCircle &rhs)

8.28 packing/EuclidPacking.cpp File Reference

```
#include "EuclidPacking.hpp"
```

8.29 packing/EuclidPacking.hpp File Reference

```
#include <QHash>
#include <memory>
#include "graph/Graph.hpp"
#include "packing/Packing.hpp"
```

Classes

- class Circles::Packing::EuclidPacking

Namespaces

- Circles
- Circles::Packing

Functions

- bool Circles::Packing::operator== (const EuclidPacking &lhs, const EuclidPacking &rhs)

8.30 packing/HyperCircle.cpp File Reference

```
#include <cmath>
#include "HyperCircle.hpp"
```

8.31 packing/HyperCircle.hpp File Reference

```
#include "packing/Circle.hpp"
```

Classes

- class Circles::Packing::HyperCircle

Namespaces

- Circles
- Circles::Packing

Functions

- bool Circles::Packing::operator== (const HyperCircle &lhs, const HyperCircle &rhs)

8.32 PFile.cpp File Reference

```
#include "PFile.hpp"
#include <QFile>
#include <QTextStream>
#include <QMessageBox>
#include <QStringList>
#include <QRegExp>
#include <QDebug>
```

8.33 PFile.hpp File Reference

```
#include <QWidget>
#include "Node.hpp"
#include "graphics/Packing.hpp"
```

Classes

- class PFile

8.34 README.md File Reference

8.35 ui/MainWindow.cpp File Reference

```
#include "MainWindow.hpp"
#include "ui_mainwindow.h"
#include <QDebug>
#include <QtMath>
#include <QWidget>
#include <QFileDialog>
#include "../PFile.hpp"
#include "../graphics/SelectionPacking.hpp"
#include "../Node.hpp"
#include "PackingView.hpp"
```

Macros

- #define PI 3.141592653589793238463

8.35.1 Macro Definition Documentation

8.35.1.1 #define PI 3.141592653589793238463

8.36 ui/MainWindow.hpp File Reference

```
#include <QMainWindow>
#include <QWidget>
#include <QGraphicsScene>
#include "../graphics/Packing.hpp"
#include "ShapeSelector.hpp"
```

Classes

- class MainWindow

Namespaces

- Ui

8.37 ui/PackingView.cpp File Reference

```
#include "PackingView.hpp"
#include "ui_PackingView.h"
#include <QDebug>
#include <QtMath>
#include <QWidget>
#include <QFileDialog>
#include <QGLWidget>
```

8.38 ui/PackingView.hpp File Reference

```
#include <QWidget>
#include <QGraphicsScene>
#include "../graphics/Packing.hpp"
#include "PropertyWindow.hpp"
```

Classes

- class PackingView

Namespaces

- Ui

8.39 ui/PropertyWindow.cpp File Reference

```
#include "PropertyWindow.hpp"
#include "ui_PropertyWindow.h"
#include <QDebug>
```

8.40 ui/PropertyWindow.hpp File Reference

```
#include <QWidget>
#include <QDialog>
#include "Node.hpp"
```

Classes

- class PropertyWindow

Namespaces

- Ui

8.41 ui/ShapeSelector.cpp File Reference

```
#include "ShapeSelector.hpp"
#include "ui_ShapeSelector.h"
#include <typeinfo>
#include <QList>
#include <QRectF>
#include <QMessageBox>
#include <QGLWidget>
#include <QDebug>
#include <QGraphicsEllipseItem>
#include "../Node.hpp"
#include "../graphics/Packing.hpp"
```

8.42 ui/ShapeSelector.hpp File Reference

```
#include <QWidget>
#include <QDialog>
#include <QResizeEvent>
#include <QList>
#include <QPointF>
#include <QGraphicsPolygonItem>
#include <QPolygonF>
#include "../graphics/Packing.hpp"
#include "../graphics/SelectionVertex.hpp"
```

Classes

- class ShapeSelector

Namespaces

- Ui

Index

_center
    Circles::Packing::Circle, 17
_circles
    Circles::Packing::Packing, 42
_graph
    Circles::Packing::Packing, 42
_index
    Circles::Packing::Circle, 17
_radius
    Circles::Packing::Circle, 17
~MainWindow
    MainWindow, 31
~Node
    Node, 33
~Packing
    Packing, 37
~PackingView
    PackingView, 43
~PropertyWindow
    PropertyWindow, 44
~ShapeSelector
    ShapeSelector, 48

Add
    SelectionPacking, 45
addCircle
    Packing, 37
addEdge
    Circles::Graph::Graph, 26
addNeibhour
    Node, 33
addNode
    Packing, 37
addNode_fast
    Packing, 37
addToSelection
    SelectionPacking, 46
angle
    Circles::Packing::Packing, 41
    Packing, 37
angle_euclidean
    Packing, 37
angle_hyperbolic
    Packing, 37
anglesum
    Circles::Packing::Packing, 41
    Packing, 37

bHasPosition
    Node, 34
Boundary, 13
    Boundary, 13
    boundingRect, 13
    paint, 13
boundary
    Circles::Graph::Graph, 26
Boundary.cpp
    PI, 52
boundaryNodes
    Packing, 40
boundingRect
    Boundary, 13
    Circle, 14
    Connector, 18
    SelectionVertex, 47

center
    Circles::Packing::Circle, 15
    Circles::Packing::EuclidCircle, 20
    Circles::Packing::HyperCircle, 29
centerCircleID
    Packing, 40
Circle, 13
    boundingRect, 14
    Circle, 14
    Circles::Packing::Circle, 15
    getNode, 14
    getSelectionState, 14
    None, 14
    paint, 14
    Selected, 14
    SelectionState, 14
    setSelectionState, 14
    shape, 15
    Surrounded, 14
circle
    Circles::Packing::Packing, 41
Circles, 11
circles
    Circles::Packing::Packing, 42
    Packing, 40
Circles::Graph, 11
    Node, 11
    operator==, 11
Circles::Graph::Edge, 18
    Edge, 19
    getX, 19
    getY, 19
    operator=, 19
    operator==, 19

set, 19
setX, 19
setY, 19
Circles::Graph::Graph, 25
addEdge, 26
boundary, 26
getEdges, 26
getNodes, 26
Graph, 26
hasEdge, 27
hasFullFlower, 27
isBoundary, 27
neighbours, 27
operator=, 27, 28
removeEdge, 28
sortedNeighbours, 28
Circles::Packing, 11
operator<, 12
operator==, 12
Circles::Packing::Circle, 15
_center, 17
_index, 17
_radius, 17
center, 15
Circle, 15
index, 16
operator<, 17
projCenter, 16
projRadius, 16
radius, 16
setCenter, 16
setIndex, 17
setRadius, 17
Circles::Packing::EuclidCircle, 19
center, 20
EuclidCircle, 20
operator=, 21
operator==, 23
projCenter, 21
projRadius, 21
radius, 21
setCenter, 21
setRadius, 21
Circles::Packing::EuclidPacking, 23
EuclidPacking, 23, 25
operator=, 25
operator==, 25
Circles::Packing::HyperCircle, 28
center, 29
HyperCircle, 29
operator=, 29
operator==, 30
projCenter, 29
projRadius, 29
radius, 30
setCenter, 30
setRadius, 30
Circles::Packing::Packing, 40
_circles, 42
_graph, 42
angle, 41
anglesum, 41
circle, 41
circles, 42
graph, 42
layout, 42
repack, 42
clearSelection
SelectionPacking, 46
color
Node, 34
Connector, 18
boundingRect, 18
Connector, 18
paint, 18
connectors
Packing, 40

delNeibhour
Node, 33
delNode
Packing, 38
delNode_fast
Packing, 38
delPosition
Node, 33
drawCenters
Packing, 40
drawCircles
Packing, 40
drawForeground
Packing, 38
drawIndicies
Packing, 40
drawLinks
Packing, 40

Edge
Circles::Graph::Edge, 19
EuclidCircle
Circles::Packing::EuclidCircle, 20
EuclidPacking
Circles::Packing::EuclidPacking, 23, 25
EuclideanPacking
graphics/Packing.hpp, 55

generateHexArray
Node, 33
generatePacking
PFile, 44
getColor
Node, 33
getDrawCenters
Packing, 38
getDrawCircles
Packing, 38
getDrawIndicies

Packing, 38
getDrawLinks
Packing, 38
getEdges
Circles::Graph::Graph, 26
getId
Node, 33
getNeibhourCount
Node, 33
getNeibhours
Node, 33
getNode
Circle, 14
getNodes
Circles::Graph::Graph, 26
Packing, 38
getPosition
Node, 33
getRadius
Node, 33
getSelectionState
Circle, 14
getType
Packing, 38
getX
Circles::Graph::Edge, 19
getY
Circles::Graph::Edge, 19
gotClick
ShapeSelectorGraphicsView, 49
Graph
Circles::Graph::Graph, 26
graph
Circles::Packing::Packing, 42
graph/Edge.cpp, 51
graph/Edge.hpp, 51
graph/Graph.cpp, 51
graph/Graph.hpp, 51
graphics/Boundary.cpp, 52
graphics/Boundary.hpp, 52
graphics/Circle.cpp, 52
graphics/Circle.hpp, 53
graphics/Connector.cpp, 53
graphics/Connector.hpp, 54
graphics/Packing.cpp, 54
PI, 54
graphics/Packing.hpp, 54
EuclideanPacking, 55
HyperbolicPacking, 55
PackingType, 55
graphics/SelectionPacking.cpp, 55
graphics/SelectionPacking.hpp, 56
graphics/SelectionVertex.cpp, 56
graphics/SelectionVertex.hpp, 56
graphics/ShapeSelectorGraphicsView.cpp, 56
graphics/ShapeSelectorGraphicsView.hpp, 56

hasEdge
Circles::Graph::Graph, 27
hasFullFlower
Circles::Graph::Graph, 27
Node, 33
hasHeightForWidth
ShapeSelectorGraphicsView, 49
hasPosition
Node, 33
heightForWidth
ShapeSelectorGraphicsView, 49
HyperCircle
Circles::Packing::HyperCircle, 29
HyperbolicPacking
graphics/Packing.hpp, 55

id
Node, 34
index
Circles::Packing::Circle, 16
isBoundary
Circles::Graph::Graph, 27
isExterior
Packing, 38
isInSelection
SelectionPacking, 46
isInterior
Packing, 38
isNeibhour
Node, 34

layout
Circles::Packing::Packing, 42
Packing, 38
layout_euclidean
Packing, 39
layout_hyperbolic
Packing, 39

main
main.cpp, 57
main.cpp, 57
main, 57
MainWindow, 31
∼MainWindow, 31
MainWindow, 31
MainWindow.cpp
PI, 60
MouseMode
SelectionPacking, 45
mouseMoveEvent
SelectionPacking, 46
mousePressEvent
Packing, 39
SelectionPacking, 46
ShapeSelectorGraphicsView, 49
mouseReleaseEvent
SelectionPacking, 46

neibhours
Node, 34

neighbours
Circles::Graph::Graph, 27
newNodeSelected
Packing, 39
Node, 31
~Node, 33
addNeibhour, 33
bHasPosition, 34
Circles::Graph, 11
color, 34
delNeibhour, 33
delPosition, 33
generateHexArray, 33
getColor, 33
getId, 33
getNeibhourCount, 33
getNeibhours, 33
getPosition, 33
getRadius, 33
hasFullFlower, 33
hasPosition, 33
id, 34
isNeibhour, 34
neibhours, 34
Node, 32
position, 34
purgeNeibhours, 34
radius, 34
setColor, 34
setId, 34
setPosition, 34
setRadius, 34
sortNeibhours, 34
sortedNeibhours, 34
Node.cpp, 57
PI, 57
Node.hpp, 57
nodes
Packing, 40
None
Circle, 14
SelectionPacking, 45

operator<
Circles::Packing, 12
Circles::Packing::Circle, 17
operator=
Circles::Graph::Edge, 19
Circles::Graph::Graph, 27, 28
Circles::Packing::EuclidCircle, 21
Circles::Packing::EuclidPacking, 25
Circles::Packing::HyperCircle, 29
operator==
Circles::Graph, 11
Circles::Graph::Edge, 19
Circles::Packing, 12
Circles::Packing::EuclidCircle, 23
Circles::Packing::EuclidPacking, 25
Circles::Packing::HyperCircle, 30
PFile, 43
generatePacking, 44
PFile, 44
PFile.cpp, 59
PFile.hpp, 59
PI
Boundary.cpp, 52
graphics/Packing.cpp, 54
MainWindow.cpp, 60
Node.cpp, 57
Packing, 34
~Packing, 37
addCircle, 37
addNode, 37
addNode_fast, 37
angle, 37
angle_euclidean, 37
angle_hyperbolic, 37
anglesum, 37
boundaryNodes, 40
centerCircleID, 40
circles, 40
connectors, 40
delNode, 38
delNode_fast, 38
drawCenters, 40
drawCircles, 40
drawForeground, 38
drawIndicies, 40
drawLinks, 40
getDrawCenters, 38
getDrawCircles, 38
getDrawIndicies, 38
getDrawLinks, 38
getNodes, 38
getType, 38
isExterior, 38
isInterior, 38
layout, 38
layout_euclidean, 39
layout_hyperbolic, 39
mousePressEvent, 39
newNodeSelected, 39
nodes, 40
Packing, 36
purgeCircles, 39
recomputeConnectors, 39
refreshCircles, 39
repack, 39
resetIds, 39
selectedCircle, 40
setDrawBoundary, 39
setDrawCenters, 39
setDrawCircles, 39
setDrawIndicies, 39
setDrawLinks, 39
setPackingType, 39
type, 40

packing/Circle.cpp, 53
packing/Circle.hpp, 53
packing/EuclidCircle.cpp, 57
packing/EuclidCircle.hpp, 58
packing/EuclidPacking.cpp, 58
packing/EuclidPacking.hpp, 58
packing/HyperCircle.cpp, 58
packing/HyperCircle.hpp, 59
packing/Packing.cpp, 54
packing/Packing.hpp, 55
packingAccepted
    ShapeSelector, 48
PackingType
    graphics/Packing.hpp, 55
PackingView, 43
    ~PackingView, 43
    PackingView, 43
    setPacking, 43
paint
    Boundary, 13
    Circle, 14
    Connector, 18
    SelectionVertex, 47
position
    Node, 34
projCenter
    Circles::Packing::Circle, 16
    Circles::Packing::EuclidCircle, 21
    Circles::Packing::HyperCircle, 29
projRadius
    Circles::Packing::Circle, 16
    Circles::Packing::EuclidCircle, 21
    Circles::Packing::HyperCircle, 29
PropertyWindow, 44
    ~PropertyWindow, 44
    PropertyWindow, 44
    refresh, 44
    setNode, 44
purgeCircles
    Packing, 39
purgeNeibhours
    Node, 34

README.md, 59
radius
    Circles::Packing::Circle, 16
    Circles::Packing::EuclidCircle, 21
    Circles::Packing::HyperCircle, 30
    Node, 34
recomputeConnectors
    Packing, 39
refresh
    PropertyWindow, 44
refreshCircles
    Packing, 39
removeEdge
    Circles::Graph::Graph, 28
removeFromSelection
    SelectionPacking, 46
repack
    Circles::Packing::Packing, 42
    Packing, 39
resetIds
    Packing, 39
resizeEvent
    ShapeSelectorGraphicsView, 49

Selected
    Circle, 14
selectedCircle
    Packing, 40
SelectionPacking, 45
    Add, 45
    addToSelection, 46
    clearSelection, 46
    isInSelection, 46
    MouseMode, 45
    mouseMoveEvent, 46
    mousePressEvent, 46
    mouseReleaseEvent, 46
    None, 45
    removeFromSelection, 46
    SelectionPacking, 46
    Subtract, 45
SelectionState
    Circle, 14
SelectionVertex, 46
    boundingRect, 47
    paint, 47
    SelectionVertex, 47
    size, 47
    thickness, 47
set
    Circles::Graph::Edge, 19
setCenter
    Circles::Packing::Circle, 16
    Circles::Packing::EuclidCircle, 21
    Circles::Packing::HyperCircle, 30
setColor
    Node, 34
setDrawBoundary
    Packing, 39
setDrawCenters
    Packing, 39
setDrawCircles
    Packing, 39
setDrawIndicies
    Packing, 39
setDrawLinks
    Packing, 39
setId
    Node, 34
setIndex
    Circles::Packing::Circle, 17
setNode
    PropertyWindow, 44
setPacking
    PackingView, 43

setPackingType
Packing, 39
setPosition
Node, 34
setRadius
Circles::Packing::Circle, 17
Circles::Packing::EuclidCircle, 21
Circles::Packing::HyperCircle, 30
Node, 34
setSelectionState
Circle, 14
setX
Circles::Graph::Edge, 19
setY
Circles::Graph::Edge, 19
shape
Circle, 15
ShapeSelector, 47
∼ShapeSelector, 48
packingAccepted, 48
ShapeSelector, 48
ShapeSelectorGraphicsView, 48
gotClick, 49
hasHeightForWidth, 49
heightForWidth, 49
mousePressEvent, 49
resizeEvent, 49
ShapeSelectorGraphicsView, 49
size
SelectionVertex, 47
sortNeibhours
Node, 34
sortedNeibhours
Node, 34
sortedNeighbours
Circles::Graph::Graph, 28
Subtract
SelectionPacking, 45
Surrounded
Circle, 14

thickness
SelectionVertex, 47
type
Packing, 40

Ui, 12
ui/MainWindow.cpp, 60
ui/MainWindow.hpp, 60
ui/PackingView.cpp, 60
ui/PackingView.hpp, 61
ui/PropertyWindow.cpp, 61
ui/PropertyWindow.hpp, 61
ui/ShapeSelector.cpp, 62
ui/ShapeSelector.hpp, 62