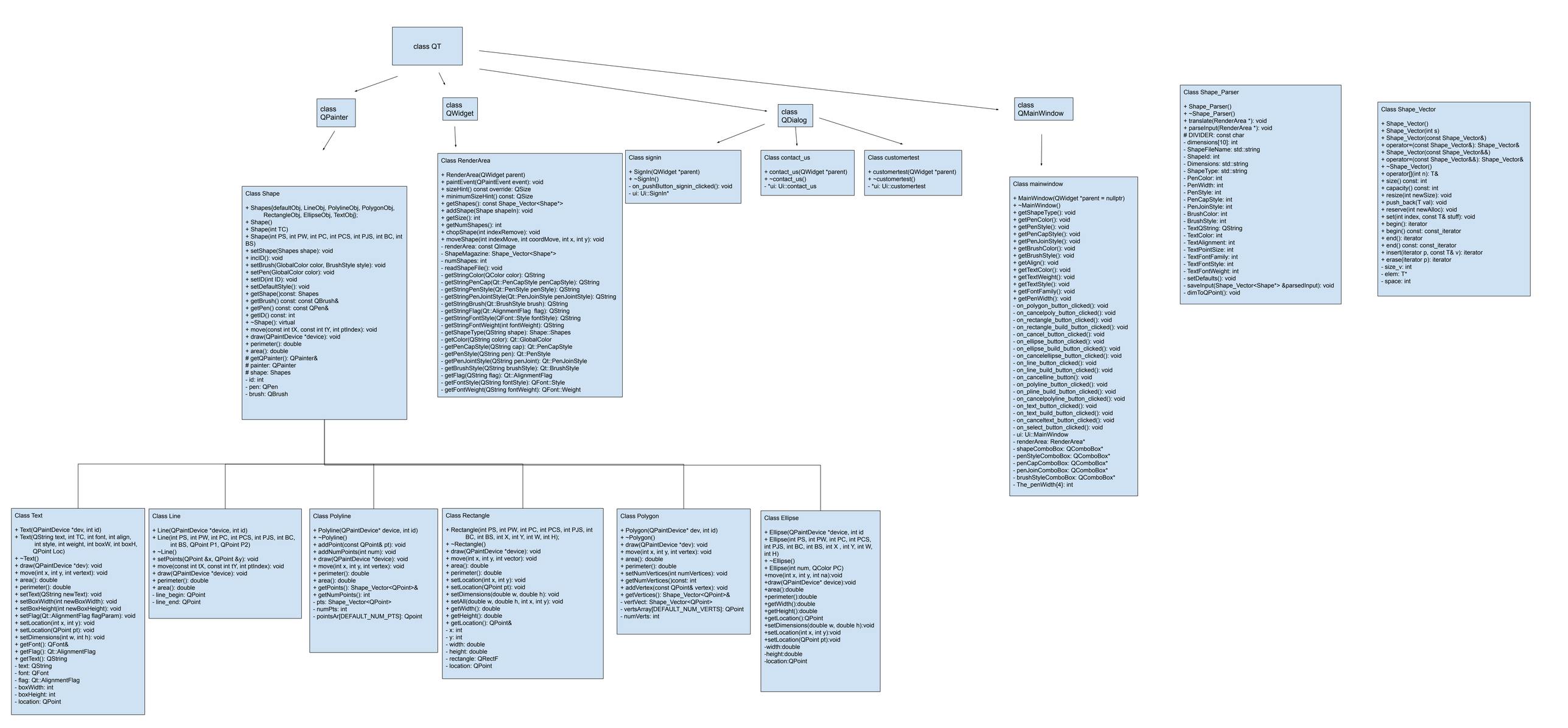
class QT
class QPainter
class QWidget
class QDialog
class QMainWindow
Class Shape
+ Shapes{defaultObj, LineObj, PolylineObj, PolygonObj, RectangleObj, EllipseObj, TextObj};
+ Shape()
+ Shape(int TC)
+ Shape(int PS, int PW, int PC, int PCS, int PJS, int BC, int BS)
+ setShape(Shapes shape): void
+ incID(): void
+ setBrush(GlobalColor color, BrushStyle style): void
+ setPen(GlobalColor color): void
+ setID(int ID): void
+ setDefaultStyle(): void
+ getShape()const: Shapes
+ getBrush() const: const QBrush&
+ getPen() const: const QPen&
+ getID() const: int
+ ~Shape(): virtual
+ move(const int tX, const int tY, int ptIndex): void
+ draw(QPaintDevice *device): void
+ perimeter(): double
+ area(): double
# getQPainter(): QPainter&
# painter: QPainter
# shape: Shapes
- id: int
- pen: QPen
- brush: QBrush
Class RenderArea
+ RenderArea(QWidget parent)
+ paintEvent(QPaintEvent event): void
+ sizeHint() const override: QSize
+ minimumSizeHint() const: QSize
+ getShapes(): const Shape_Vector<Shape*>
+ addShape(Shape shapeIn): void
+ getSize(): int
+ getNumShapes(): int
+ chopShape(int indexRemove): void
+ moveShape(int indexMove, int coordMove, int x, int y): void
- renderArea: const QImage
- ShapeMagazine: Shape_Vector<Shape*>
- numShapes: int
- readShapeFile(): void
- getStringColor(QColor color): QString
- getStringPenCap(Qt::PenCapStyle penCapStyle): QString
- getStringPenStyle(Qt::PenStyle penStyle): QString
- getStringPenJoinStyle(Qt::PenJoinStyle penJoinStyle): QString
- getStringBrush(Qt::BrushStyle brush): QString
- getStringFlag(Qt::AlignmentFlag flag): QString
- getStringFontStyle(QFont::Style fontStyle): QString
- getStringFontWeight(int fontWeight): QString
- getShapeType(QString shape): Shape::Shapes
- getColor(QString color): Qt::GlobalColor
- getPenCapStyle(QString cap): Qt::PenCapStyle
- getPenStyle(QString pen): Qt::PenStyle
- getPenJoinStyle(QString penJoint): Qt::PenJoinStyle
- getBrushStyle(QString brushStyle): Qt::BrushStyle
- getFlag(QString flag): Qt::AlignmentFlag
- getFontStyle(QString fontStyle): QFont::Style
- getFontWeight(QString fontWeight): QFont::Weight
Class signin
+ SignIn(QWidget *parent)
+ ~SignIn()
- on_pushButton_signin_clicked(): void
- ui: Ui::SignIn*
Class contact_us
+ contact_us(QWidget *parent)
+ ~contact_us()
- *ui: Ui::contact_us
Class customertest
+ customertest(QWidget *parent)
+ ~customertest()
- *ui: Ui::customertest
Class mainwindow
+ MainWindow(QWidget *parent = nullptr)
+ ~MainWindow()
+ getShapeType(): void
+ getPenColor(): void
+ getPenStyle(): void
+ getPenCapStyle(): void
+ getPenJoinStyle(): void
+ getBrushColor(): void
+ getBrushStyle(): void
+ getAlign(): void
+ getTextColor(): void
+ getTextWeight(): void
+ getTextStyle(): void
+ getFontFamily(): void
+ getPenWidth(): void
- on_polygon_button_clicked(): void
- on_cancelpoly_button_clicked(): void
- on_rectangle_button_clicked(): void
- on_rectangle_build_button_clicked(): void
- on_cancel_button_clicked(): void
- on_ellipse_button_clicked(): void
- on_ellipse_build_button_clicked(): void
- on_cancelellipse_button_clicked(): void
- on_line_button_clicked(): void
- on_line_build_button_clicked(): void
- on_cancelline_button(): void
- on_polyline_button_clicked(): void
- on_pline_build_button_clicked(): void
- on_cancelpolyline_button_clicked(): void
- on_text_button_clicked(): void
- on_text_build_button_clicked(): void
- on_canceltext_button_clicked(): void
- on_select_button_clicked(): void
- ui: Ui::MainWindow
- renderArea: RenderArea*
- shapeComboBox: QComboBox*
- penStyleComboBox: QComboBox*
- penCapComboBox: QComboBox*
- penJoinComboBox: QComboBox*
- brushStyleComboBox: QComboBox*
- The_penWidth{4}: int
Class Shape_Parser
+ Shape_Parser()
+ ~Shape_Parser()
+ translate(RenderArea *): void
+ parseInput(RenderArea *): void
# DIVIDER: const char
- dimensions[10]: int
- ShapeFileName: std::string
- ShapeId: int
- Dimensions: std::string
- ShapeType: std::string
- PenColor: int
- PenWidth: int
- PenStyle: int
- PenCapStyle: int
- PenJoinStyle: int
- BrushColor: int
- BrushStyle: int
- TextQString: QString
- TextColor: int
- TextAlignment: int
- TextPointSize: int
- TextFontFamily: int
- TextFontStyle: int
- TextFontWeight: int
- setDefaults(): void
- saveInput(Shape_Vector<Shape*> &parsedInput): void
- dimToQPoint(): void
Class Shape_Vector
+ Shape_Vector()
+ Shape_Vector(int s)
+ Shape_Vector(const Shape_Vector&)
+ operator=(const Shape_Vector&): Shape_Vector&
+ Shape_Vector(const Shape_Vector&&)
+ operator=(const Shape_Vector&&): Shape_Vector&
+ ~Shape_Vector()
+ operator[](int n): T&
+ size() const: int
+ capacity() const: int
+ resize(int newSize): void
+ push_back(T val): void
+ reserve(int newAlloc): void
+ set(int index, const T& stuff): void
+ begin(): iterator
+ begin() const: const_iterator
+ end(): iterator
+ end() const: const_iterator
+ insert(iterator p, const T& v): iterator
+ erase(iterator p): iterator
- size_v: int
- elem: T*
- space: int
Class Text
+ Text(QPaintDevice *dev, int id)
+ Text(QString text, int TC, int font, int align, int style, int weight, int boxW, int boxH, QPoint Loc)
+ ~Text()
+ draw(QPaintDevice *dev): void
+ move(int x, int y, int vertext): void
+ area(): double
+ perimeter(): double
+ setText(QString newText): void
+ setBoxWidth(int newBoxWidth): void
+ setBoxHeight(int newBoxHeight): void
+ setFlag(Qt::AlignmentFlag flagParam): void
+ setLocation(int x, int y): void
+ setLocation(QPoint pt): void
+ setDimensions(int w, int h): void
+ getFont(): QFont&
+ getFlag(): Qt::AlignmentFlag
+ getText(): QString
- text: QString
- font: QFont
- flag: Qt::AlignmentFlag
- boxWidth: int
- boxHeight: int
- location: QPoint
Class Line
+ Line(QPaintDevice *device, int id)
+ Line(int PS, int PW, int PC, int PCS, int PJS, int BC, int BS, QPoint P1, QPoint P2)
+ ~Line()
+ setPoints(QPoint &x, QPoint &y): void
+ move(const int tX, const int tY, int ptIndex): void
+ draw(QPaintDevice *device): void
+ perimeter(): double
+ area(): double
- line_begin: QPoint
- line_end: QPoint
Class Polyline
+ Polyline(QPaintDevice* device, int id)
+ ~Polyline()
+ addPoint(const QPoint& pt): void
+ addNumPoints(int num): void
+ draw(QPaintDevice *device): void
+ move(int x, int y, int vertex): void
+ perimeter(): double
+ area(): double
+ getPoints(): Shape_Vector<QPoint>&
+ getNumPoints(): int
- pts: Shape_Vector<QPoint>
- numPts: int
- pointsAr[DEFAULT_NUM_PTS]: Qpoint
Class Rectangle
+ Rectangle(int PS, int PW, int PC, int PCS, int PJS, int BC, int BS, int X, int Y, int W, int H);
+ ~Rectangle()
+ draw(QPaintDevice *device): void
+ move(int x, int y, int vector): void
+ area(): double
+ perimeter(): double
+ setLocation(int x, int y): void
+ setLocation(QPoint pt): void
+ setDimensions(double w, double h): void
+ setAll(double w, double h, int x, int y): void
+ getWidth(): double
+ getHeight(): double
+ getLocation(): QPoint&
- x: int
- y: int
- width: double
- height: double
- rectangle: QRectF
- location: QPoint
Class Polygon
+ Polygon(QPaintDevice* dev, int id)
+ ~Polygon()
+ draw(QPaintDevice *dev): void
+ move(int x, int y, int vertex): void
+ area(): double
+ perimeter(): double
+ setNumVertices(int numVertices): void
+ getNumVertices()const: int
+ addVertex(const QPoint& vertex): void
+ getVertices(): Shape_Vector<QPoint>&
- vertVect: Shape_Vector<QPoint>
- vertsArray[DEFAULT_NUM_VERTS]: QPoint
- numVerts: int
Class Ellipse
+ Ellipse(QPaintDevice *device, int id
+ Ellipse(int PS, int PW, int PC, int PCS, int PJS, int BC, int BS, int X , int Y, int W, int H)
+ ~Ellipse()
+ Ellipse(int num, QColor PC)
+move(int x, int y, int na):void
+draw(QPaintDevice* device):void
+area():double
+perimeter():double
+getWidth():double
+getHeight():double
+getLocation():QPoint
+setDimensions(double w, double h):void
+setLocation(int x, int y):void
+setLocation(QPoint pt):void
-width:double
-height:double
-location:QPoint