2D graphics modeler

Generated by Doxygen 1.13.2

1 Topic Index 1
1.1 Topics . . . 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 Topic Documentation 11
6.1 > Shapes . . . 11

7 Namespace Documentation 13
7.1 alpha Namespace Reference . . . 13
7.2 QT_WARNING_DISABLE_DEPRECATED Namespace Reference . . . 13
7.3 Ui Namespace Reference . . . 13

8 Class Documentation 15
8.1 ApiClient Class Reference . . . 15
8.1.1 Detailed Description . . . 17
8.1.2 Constructor & Destructor Documentation . . . 17
8.1.2.1 ApiClient() . . . 17
8.1.3 Member Function Documentation . . . 17
8.1.3.1 BadDeleteReply . . . 17
8.1.3.2 BadGetReply . . . 18
8.1.3.3 BadPostReply . . . 19
8.1.3.4 DeleteRenderAreaAll() . . . 19
8.1.3.5 DeleteShapesAll() . . . 19
8.1.3.6 DeleteTestimonialsAll() . . . 19
8.1.3.7 DeleteUsersAll() . . . 19
8.1.3.8 GetRenderArea() . . . 20
8.1.3.9 GetShapes() . . . 20
8.1.3.10 GetTestimonials() . . . 20
8.1.3.11 GetUsers() . . . 20
8.1.3.12 GoodDeleteReply . . . 20
8.1.3.13 GoodGetReply . . . 20
8.1.3.14 GoodPostReply . . . 21
8.1.3.15 PostRenderArea() . . . 21
8.1.3.16 PostShapes() . . . 22

8.1.3.17 PostTestimonials() . . . 22
8.1.3.18 PostUsers() . . . 22
8.2 AppDriver Class Reference . . . 22
8.2.1 Constructor & Destructor Documentation . . . 24
8.2.1.1 AppDriver() . . . 24
8.2.1.2 ~AppDriver() . . . 24
8.2.2 Member Function Documentation . . . 24
8.2.2.1 loadAllData() . . . 24
8.2.2.2 run() . . . 24
8.2.2.3 shutdown() . . . 25
8.3 Circle Class Reference . . . 25
8.3.1 Detailed Description . . . 29
8.3.2 Constructor & Destructor Documentation . . . 29
8.3.2.1 Circle() . . . 29
8.3.3 Member Function Documentation . . . 30
8.3.3.1 Area() . . . 30
8.3.3.2 Draw() . . . 30
8.3.3.3 getR() . . . 31
8.3.3.4 isPointInside() . . . 31
8.3.3.5 Perimeter() . . . 32
8.3.3.6 setR() . . . 32
8.3.3.7 setX() . . . 33
8.3.3.8 setY() . . . 33
8.4 ColumnEditDelegate Class Reference . . . 34
8.4.1 Constructor & Destructor Documentation . . . 35
8.4.1.1 ColumnEditDelegate() . . . 35
8.4.2 Member Function Documentation . . . 35
8.4.2.1 createEditor() . . . 35
8.4.2.2 setCanEdit() . . . 36
8.5 Ellipse Class Reference . . . 36
8.5.1 Detailed Description . . . 40
8.5.2 Constructor & Destructor Documentation . . . 40
8.5.2.1 Ellipse() . . . 40
8.5.3 Member Function Documentation . . . 41
8.5.3.1 Area() . . . 41
8.5.3.2 Draw() . . . 41
8.5.3.3 getA() . . . 42
8.5.3.4 getB() . . . 42
8.5.3.5 isPointInside() . . . 42
8.5.3.6 Perimeter() . . . 43
8.5.3.7 setA() . . . 43
8.5.3.8 setB() . . . 44

8.5.3.9 setX() . . . 44
8.5.3.10 setY() . . . 45
8.6 Line Class Reference . . . 45
8.6.1 Detailed Description . . . 49
8.6.2 Constructor & Destructor Documentation . . . 49
8.6.2.1 Line() . . . 49
8.6.3 Member Function Documentation . . . 50
8.6.3.1 Area() . . . 50
8.6.3.2 Draw() . . . 50
8.6.3.3 getEndPoint() . . . 51
8.6.3.4 getStartPoint() . . . 52
8.6.3.5 isPointInside() . . . 52
8.6.3.6 Move() . . . 52
8.6.3.7 Perimeter() . . . 53
8.6.3.8 setEndPoint() . . . 54
8.6.3.9 setStartPoint() . . . 54
8.6.3.10 setX() . . . 54
8.6.3.11 setY() . . . 55
8.7 LoginWindow Class Reference . . . 56
8.7.1 Constructor & Destructor Documentation . . . 57
8.7.1.1 LoginWindow() . . . 57
8.7.2 Member Function Documentation . . . 57
8.7.2.1 loginRequested . . . 57
8.7.2.2 password() . . . 58
8.7.2.3 signupRequested . . . 58
8.7.2.4 username() . . . 58
8.8 MainWindow Class Reference . . . 59
8.8.1 Constructor & Destructor Documentation . . . 61
8.8.1.1 MainWindow() [1/2] . . . 61
8.8.1.2 MainWindow() [2/2] . . . 62
8.8.1.3 ~MainWindow() . . . 62
8.8.2 Member Function Documentation . . . 62
8.8.2.1 deleteAllShapes . . . 62
8.8.2.2 displayedTextChanged . . . 62
8.8.2.3 drawShapes() . . . 62
8.8.2.4 loginAttempt . . . 62
8.8.2.5 loginFailed . . . 63
8.8.2.6 loginSuccess . . . 63
8.8.2.7 newUserAdded . . . 63
8.8.2.8 onLoginClicked . . . 64
8.8.2.9 onLoginRequest . . . 64
8.8.2.10 onRenderAreaChanged . . . 65

8.8.2.11 onRenderAreaNotChanged . . . 65
8.8.2.12 onSignupRequest . . . 66
8.8.2.13 onUserAuthentication . . . 66
8.8.2.14 onUserAuthenticationFailure . . . 67
8.8.2.15 shapeAdded . . . 68
8.8.2.16 shapeChanged . . . 68
8.8.2.17 shapeDeleted . . . 68
8.8.2.18 shapes_to_treeWidget() . . . 68
8.8.2.19 showRenderStatusMessage . . . 70
8.9 Parser Class Reference . . . 70
8.9.1 Constructor & Destructor Documentation . . . 72
8.9.1.1 Parser() [1/3] . . . 72
8.9.1.2 ~Parser() . . . 72
8.9.1.3 Parser() [2/3] . . . 72
8.9.1.4 Parser() [3/3] . . . 73
8.9.2 Member Function Documentation . . . 73
8.9.2.1 JsonToShapes() . . . 73
8.9.2.2 JsonToTestimonials() . . . 73
8.9.2.3 JsonToUsers() . . . 74
8.9.2.4 operator=() [1/2] . . . 74
8.9.2.5 operator=() [2/2] . . . 74
8.9.2.6 PrintShapeVector() . . . 74
8.9.2.7 ShapesToJson() . . . 75
8.9.2.8 TestimonialsToJson() . . . 75
8.9.2.9 UsersToJson() . . . 76
8.10 Polygon Class Reference . . . 76
8.10.1 Detailed Description . . . 80
8.10.2 Constructor & Destructor Documentation . . . 80
8.10.2.1 Polygon() . . . 80
8.10.3 Member Function Documentation . . . 81
8.10.3.1 Area() . . . 81
8.10.3.2 Draw() . . . 82
8.10.3.3 getPointsList() . . . 82
8.10.3.4 isPointInside() . . . 83
8.10.3.5 Move() . . . 83
8.10.3.6 Perimeter() . . . 84
8.10.3.7 setPointsList() . . . 85
8.10.3.8 setX() . . . 86
8.10.3.9 setY() . . . 86
8.11 Polyline Class Reference . . . 87
8.11.1 Constructor & Destructor Documentation . . . 91
8.11.1.1 Polyline() . . . 91

8.11.2 Member Function Documentation . . . 92
8.11.2.1 Area() . . . 92
8.11.2.2 Draw() . . . 92
8.11.2.3 getPointsList() . . . 93
8.11.2.4 isPointInside() . . . 93
8.11.2.5 Move() . . . 94
8.11.2.6 Perimeter() . . . 95
8.11.2.7 setPointsList() . . . 95
8.11.2.8 setX() . . . 95
8.11.2.9 setY() . . . 96
8.12 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t Struct Reference . . . 97
8.13 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t Struct Reference . . . 97
8.14 Rectangle Class Reference . . . 98
8.14.1 Constructor & Destructor Documentation . . . 101
8.14.1.1 Rectangle() . . . 101
8.14.2 Member Function Documentation . . . 102
8.14.2.1 Area() . . . 102
8.14.2.2 Draw() . . . 102
8.14.2.3 getLength() . . . 103
8.14.2.4 getWidth() . . . 103
8.14.2.5 isPointInside() . . . 103
8.14.2.6 Perimeter() . . . 104
8.14.2.7 setLength() . . . 104
8.14.2.8 setWidth() . . . 105
8.14.2.9 setX() . . . 105
8.14.2.10 setY() . . . 106
8.15 RenderArea Class Reference . . . 106
8.15.1 Constructor & Destructor Documentation . . . 109
8.15.1.1 RenderArea() . . . 109
8.15.2 Member Function Documentation . . . 109
8.15.2.1 getShapes() . . . 109
8.15.2.2 getShapeSelected() . . . 109
8.15.2.3 getShapeSelectedIndex() . . . 109
8.15.2.4 mouseDoubleClickEvent() . . . 109
8.15.2.5 mouseMoveEvent() . . . 109
8.15.2.6 mousePressEvent() . . . 110
8.15.2.7 mouseReleaseEvent() . . . 110
8.15.2.8 paintEvent() . . . 110
8.15.2.9 resetSelection() . . . 110
8.15.2.10 setEditPrivileges() . . . 111
8.15.2.11 setRenderShapes() . . . 111
8.15.2.12 setShapeSelectedIndex() . . . 111

8.15.2.13 updateShapeDisplayCoords() . . . 111
8.16 RenderAreaManager Class Reference . . . 112
8.16.1 Constructor & Destructor Documentation . . . 114
8.16.1.1 RenderAreaManager() . . . 114
8.16.1.2 ~RenderAreaManager() . . . 114
8.16.2 Member Function Documentation . . . 115
8.16.2.1 addShape() . . . 115
8.16.2.2 deleteAllShapes() . . . 115
8.16.2.3 deleteShape() . . . 116
8.16.2.4 getShapesRef() . . . 116
8.16.2.5 loadShapes() . . . 116
8.16.2.6 modifyDisplayedText() . . . 117
8.16.2.7 modifyShape() . . . 117
8.16.2.8 renderAreaChanged . . . 118
8.16.2.9 renderAreaNotChanged . . . 119
8.16.2.10 saveShapes() . . . 120
8.16.2.11 statusMessage . . . 120
8.17 Shape Interface Reference . . . 120
8.17.1 Detailed Description . . . 124
8.17.2 Constructor & Destructor Documentation . . . 124
8.17.2.1 Shape() . . . 124
8.17.2.2 ~Shape() . . . 126
8.17.3 Member Function Documentation . . . 126
8.17.3.1 allocateTrackerId() . . . 126
8.17.3.2 Area() . . . 126
8.17.3.3 CreateBrushChild() . . . 127
8.17.3.4 CreateParentItem() . . . 127
8.17.3.5 CreatePenChild() . . . 128
8.17.3.6 CreatePointsChild() . . . 129
8.17.3.7 Draw() . . . 130
8.17.3.8 getBrush() . . . 130
8.17.3.9 getBrushColor() . . . 131
8.17.3.10 getBrushItems() . . . 131
8.17.3.11 getBrushItemsEnd() . . . 132
8.17.3.12 getBrushStyle() . . . 132
8.17.3.13 getChildEnd() . . . 133
8.17.3.14 getChildItems() . . . 133
8.17.3.15 getPainter() . . . 133
8.17.3.16 getParentItem() . . . 134
8.17.3.17 getPen() . . . 134
8.17.3.18 getPenCapStyle() . . . 135
8.17.3.19 getPenColor() . . . 135

8.17.3.20 getPenItems() . . . 136
8.17.3.21 getPenItemsEnd() . . . 136
8.17.3.22 getPenJoinStyle() . . . 137
8.17.3.23 getPenStyle() . . . 137
8.17.3.24 getPenWidth() . . . 137
8.17.3.25 getPoints() . . . 138
8.17.3.26 getPointsItems() . . . 138
8.17.3.27 getSelected() . . . 138
8.17.3.28 getShapeId() . . . 139
8.17.3.29 getShapeType() . . . 140
8.17.3.30 getTrackerId() . . . 140
8.17.3.31 getX() . . . 141
8.17.3.32 getY() . . . 142
8.17.3.33 isPointInside() . . . 143
8.17.3.34 Move() . . . 144
8.17.3.35 Perimeter() . . . 145
8.17.3.36 setBrush() . . . 145
8.17.3.37 setInternalBrush() . . . 145
8.17.3.38 setInternalPen() . . . 145
8.17.3.39 setPen() . . . 146
8.17.3.40 setSelected() . . . 146
8.17.3.41 setShapeId() . . . 146
8.17.3.42 setShapeType() . . . 147
8.17.3.43 setTrackerId() . . . 147
8.17.3.44 setX() . . . 147
8.17.3.45 setY() . . . 148
8.17.4 Friends And Related Symbol Documentation . . . 148
8.17.4.1 operator< . . . 148
8.17.4.2 operator== . . . 148
8.17.5 Member Data Documentation . . . 149
8.17.5.1 brushItems . . . 149
8.17.5.2 childItems . . . 149
8.17.5.3 nextTracker . . . 149
8.17.5.4 parentItem . . . 149
8.17.5.5 penItems . . . 149
8.17.5.6 pointsItems . . . 149
8.17.5.7 trackersInUse . . . 150
8.18 ShapesManager Class Reference . . . 150
8.18.1 Constructor & Destructor Documentation . . . 152
8.18.1.1 ShapesManager() . . . 152
8.18.1.2 ~ShapesManager() . . . 152
8.18.2 Member Function Documentation . . . 152

8.18.2.1 addShape() . . . 152
8.18.2.2 deleteAllShapes() . . . 153
8.18.2.3 deleteShape() . . . 153
8.18.2.4 getShapesRef() . . . 154
8.18.2.5 loadShapes() . . . 154
8.18.2.6 modifyShape() . . . 154
8.18.2.7 saveShapes() . . . 154
8.18.2.8 shapesChanged . . . 155
8.18.2.9 shapesNotChanged . . . 155
8.18.2.10 statusMessage . . . 155
8.19 Square Class Reference . . . 156
8.19.1 Constructor & Destructor Documentation . . . 159
8.19.1.1 Square() . . . 159
8.19.2 Member Function Documentation . . . 160
8.19.2.1 Area() . . . 160
8.19.2.2 Draw() . . . 160
8.19.2.3 getLength() . . . 161
8.19.2.4 isPointInside() . . . 161
8.19.2.5 Perimeter() . . . 162
8.19.2.6 setLength() . . . 162
8.19.2.7 setX() . . . 163
8.19.2.8 setY() . . . 163
8.20 Testimonial Class Reference . . . 164
8.20.1 Constructor & Destructor Documentation . . . 165
8.20.1.1 Testimonial() . . . 165
8.20.2 Member Function Documentation . . . 165
8.20.2.1 fromJson() . . . 165
8.20.2.2 getAuthor() . . . 166
8.20.2.3 getContent() . . . 166
8.20.2.4 getTimestamp() . . . 166
8.20.2.5 isGuest() . . . 166
8.20.2.6 isSatisfactory() . . . 166
8.20.2.7 setIsSatisfactory() . . . 166
8.20.2.8 toJson() . . . 167
8.21 TestimonialDialog Class Reference . . . 167
8.21.1 Constructor & Destructor Documentation . . . 168
8.21.1.1 TestimonialDialog() . . . 168
8.22 TestimonialManager Class Reference . . . 168
8.22.1 Member Function Documentation . . . 169
8.22.1.1 addTestimonial() . . . 169
8.22.1.2 getDoNotShowAgain() . . . 170
8.22.1.3 getInstance() . . . 170

8.22.1.4 getSatisfactoryTestimonials() . . . 170
8.22.1.5 hasUserGivenTestimonial() . . . 170
8.22.1.6 setDoNotShowAgain() . . . 170
8.22.1.7 shouldPromptForTestimonial . . . 170
8.22.1.8 startTrackingTime() . . . 170
8.22.1.9 stopTrackingTime() . . . 170
8.23 TestimonialsDisplayDialog Class Reference . . . 171
8.23.1 Constructor & Destructor Documentation . . . 172
8.23.1.1 TestimonialsDisplayDialog() . . . 172
8.24 Text Class Reference . . . 172
8.24.1 Constructor & Destructor Documentation . . . 177
8.24.1.1 Text() . . . 177
8.24.2 Member Function Documentation . . . 177
8.24.2.1 Area() . . . 177
8.24.2.2 Draw() . . . 177
8.24.2.3 getFont() . . . 178
8.24.2.4 getFontStyle() . . . 179
8.24.2.5 getFontWeight() . . . 179
8.24.2.6 getLength() . . . 179
8.24.2.7 getTextAlignment() . . . 179
8.24.2.8 getTextColor() . . . 180
8.24.2.9 getTextString() . . . 180
8.24.2.10 getWidth() . . . 180
8.24.2.11 isPointInside() . . . 180
8.24.2.12 Perimeter() . . . 181
8.24.2.13 setAlignment() . . . 181
8.24.2.14 setInternalFont() . . . 181
8.24.2.15 setLength() . . . 182
8.24.2.16 setText() . . . 182
8.24.2.17 setWidth() . . . 182
8.24.2.18 setX() . . . 183
8.24.2.19 setY() . . . 183
8.25 UserAccount Class Reference . . . 184
8.25.1 Constructor & Destructor Documentation . . . 185
8.25.1.1 UserAccount() [1/4] . . . 185
8.25.1.2 UserAccount() [2/4] . . . 185
8.25.1.3 ~UserAccount() . . . 186
8.25.1.4 UserAccount() [3/4] . . . 186
8.25.1.5 UserAccount() [4/4] . . . 186
8.25.2 Member Function Documentation . . . 186
8.25.2.1 getPassword() . . . 186
8.25.2.2 getUsername() . . . 187

8.25.2.3 isAdmin() . . . 187
8.25.2.4 operator=() [1/2] . . . 187
8.25.2.5 operator=() [2/2] . . . 188
8.25.2.6 setAdmin() . . . 188
8.25.2.7 setPassword() . . . 188
8.25.2.8 setUserAccount() . . . 188
8.25.2.9 setUsername() . . . 188
8.26 UserManager Class Reference . . . 189
8.26.1 Constructor & Destructor Documentation . . . 191
8.26.1.1 UserManager() . . . 191
8.26.1.2 ∼UserManager() . . . 191
8.26.2 Member Function Documentation . . . 192
8.26.2.1 addUser() . . . 192
8.26.2.2 authenticate() . . . 192
8.26.2.3 authenticationFailed . . . 193
8.26.2.4 deleteAllUsers() . . . 193
8.26.2.5 deleteUser() . . . 193
8.26.2.6 getCurrUserRef() . . . 193
8.26.2.7 loadUsers() . . . 193
8.26.2.8 modifyUser() . . . 194
8.26.2.9 saveUsers() . . . 194
8.26.2.10 statusMessage . . . 194
8.26.2.11 userAuthenticated . . . 195
8.26.2.12 userChanged . . . 195
8.26.2.13 userNotChanged . . . 195
8.27 alpha::vector< T > Class Template Reference . . . 196
8.27.1 Member Typedef Documentation . . . 197
8.27.1.1 const_iterator . . . 197
8.27.1.2 iterator . . . 197
8.27.2 Constructor & Destructor Documentation . . . 197
8.27.2.1 vector() [1/4] . . . 197
8.27.2.2 vector() [2/4] . . . 197
8.27.2.3 vector() [3/4] . . . 198
8.27.2.4 vector() [4/4] . . . 198
8.27.2.5 ∼vector() . . . 198
8.27.3 Member Function Documentation . . . 198
8.27.3.1 begin() [1/2] . . . 198
8.27.3.2 begin() [2/2] . . . 199
8.27.3.3 capacity() . . . 199
8.27.3.4 end() [1/2] . . . 199
8.27.3.5 end() [2/2] . . . 199
8.27.3.6 erase() . . . 199

8.27.3.7 insert() . . . 200
8.27.3.8 operator=() [1/2] . . . 200
8.27.3.9 operator=() [2/2] . . . 200
8.27.3.10 operator[]() [1/2] . . . 201
8.27.3.11 operator[]() [2/2] . . . 201
8.27.3.12 push_back() . . . 201
8.27.3.13 reserve() . . . 202
8.27.3.14 resize() . . . 202
8.27.3.15 size() . . . 202

9 File Documentation **205**

9.1 src/code/all_shapes.h File Reference . . . 205
9.1.1 Enumeration Type Documentation . . . 205
9.1.1.1 ShapeIDs . . . 205
9.2 all_shapes.h . . . 206
9.3 src/code/ApiClient.cpp File Reference . . . 206
9.4 src/code/ApiClient.h File Reference . . . 206
9.4.1 Detailed Description . . . 207
9.5 ApiClient.h . . . 207
9.6 src/code/AppDriver.cpp File Reference . . . 209
9.7 src/code/AppDriver.h File Reference . . . 209
9.8 AppDriver.h . . . 210
9.9 src/code/circle.cpp File Reference . . . 211
9.10 src/code/circle.h File Reference . . . 211
9.11 circle.h . . . 212
9.12 src/code/ColumnEditDelegate.h File Reference . . . 212
9.13 ColumnEditDelegate.h . . . 213
9.14 src/code/ellipse.cpp File Reference . . . 214
9.15 src/code/ellipse.h File Reference . . . 214
9.16 ellipse.h . . . 215
9.17 src/code/line.cpp File Reference . . . 215
9.18 src/code/line.h File Reference . . . 216
9.19 line.h . . . 216
9.20 src/code/loginwindow.cpp File Reference . . . 217
9.21 src/code/loginwindow.h File Reference . . . 217
9.22 loginwindow.h . . . 218
9.23 src/code/main.cpp File Reference . . . 219
9.23.1 Function Documentation . . . 219
9.23.1.1 main() . . . 219
9.24 src/code/mainwindow.cpp File Reference . . . 220
9.25 src/code/mainwindow.h File Reference . . . 220
9.26 mainwindow.h . . . 221

9.27 src/code/moc_ApiClient.cpp File Reference . . . 223
9.27.1 Macro Definition Documentation . . . 224
9.27.1.1 Q_CONSTINIT . . . 224
9.28 src/code/moc_UserManager.cpp File Reference . . . 224
9.28.1 Macro Definition Documentation . . . 224
9.28.1.1 Q_CONSTINIT . . . 224
9.29 src/code/Parser.cpp File Reference . . . 225
9.30 src/code/Parser.h File Reference . . . 225
9.31 Parser.h . . . 226
9.32 src/code/polygon.cpp File Reference . . . 227
9.33 src/code/polygon.h File Reference . . . 228
9.34 polygon.h . . . 228
9.35 src/code/polyline.cpp File Reference . . . 229
9.36 src/code/polyline.h File Reference . . . 229
9.37 polyline.h . . . 230
9.38 src/code/rectangle.cpp File Reference . . . 231
9.39 src/code/rectangle.h File Reference . . . 231
9.40 rectangle.h . . . 232
9.41 src/code/renderarea.cpp File Reference . . . 232
9.42 src/code/renderarea.h File Reference . . . 232
9.43 renderarea.h . . . 233
9.44 src/code/RenderAreaManager.cpp File Reference . . . 234
9.45 src/code/RenderAreaManager.h File Reference . . . 234
9.46 RenderAreaManager.h . . . 235
9.47 src/code/shape.cpp File Reference . . . 236
9.47.1 Function Documentation . . . 236
9.47.1.1 operator<() . . . 236
9.47.1.2 operator==() . . . 236
9.48 src/code/shape.h File Reference . . . 237
9.48.1 Variable Documentation . . . 238
9.48.1.1 PI . . . 238
9.49 shape.h . . . 238
9.50 src/code/ShapesManager.cpp File Reference . . . 239
9.51 src/code/ShapesManager.h File Reference . . . 240
9.52 ShapesManager.h . . . 241
9.53 src/code/square.cpp File Reference . . . 241
9.54 src/code/square.h File Reference . . . 242
9.55 square.h . . . 242
9.56 src/code/Testimonial.cpp File Reference . . . 243
9.57 src/code/Testimonial.h File Reference . . . 243
9.58 Testimonial.h . . . 244
9.59 src/code/TestimonialDialog.cpp File Reference . . . 244

9.60 src/code/TestimonialDialog.h File Reference . . . 245
9.61 TestimonialDialog.h . . . 246
9.62 src/code/TestimonialManager.cpp File Reference . . . 246
9.63 src/code/TestimonialManager.h File Reference . . . 246
9.64 TestimonialManager.h . . . 247
9.65 src/code/TestimonialsDisplayDialog.cpp File Reference . . . 248
9.66 src/code/TestimonialsDisplayDialog.h File Reference . . . 248
9.67 TestimonialsDisplayDialog.h . . . 249
9.68 src/code/text.cpp File Reference . . . 250
9.69 src/code/text.h File Reference . . . 250
9.70 text.h . . . 251
9.71 src/code/UserAccount.cpp File Reference . . . 252
9.72 src/code/UserAccount.h File Reference . . . 252
9.73 UserAccount.h . . . 253
9.74 src/code/UserManager.cpp File Reference . . . 253
9.75 src/code/UserManager.h File Reference . . . 254
9.76 UserManager.h . . . 255
9.77 src/code/vector.h File Reference . . . 255
9.77.1 Macro Definition Documentation . . . 256
9.77.1.1 ALPHA_VECTOR_H . . . 256
9.78 vector.h . . . 256
9.79 src/code/webservice.cpp File Reference . . . 260
9.79.1 Detailed Description . . . 261
9.79.2 Function Documentation . . . 261
9.79.2.1 get_json_file() . . . 261
9.79.2.2 main() . . . 262

Index **265**

Chapter 1

Topic Index

1.1 Topics

Here is a list of all topics with brief descriptions:

> Shapes 11

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

alpha 13
QT_WARNING_DISABLE_DEPRECATED 13
Ui 13

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

- Parser . . . 70
- QDialog
  - LoginWindow . . . 56
  - TestimonialDialog . . . 167
  - TestimonialsDisplayDialog . . . 171
- QMainWindow
  - MainWindow . . . 59
- QObject
  - ApiClient . . . 15
  - AppDriver . . . 22
  - RenderAreaManager . . . 112
  - ShapesManager . . . 150
  - TestimonialManager . . . 168
  - UserManager . . . 189
- QStyledItemDelegate
  - ColumnEditDelegate . . . 34
- QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t . . . 97
- QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t . . . 97
- QWidget
  - RenderArea . . . 106
- Shape . . . 120
  - Circle . . . 25
  - Ellipse . . . 36
  - Line . . . 45
  - Polygon . . . 76
  - Polyline . . . 87
  - Rectangle . . . 98
  - Square . . . 156
  - Text . . . 172
- Testimonial . . . 164
- UserAccount . . . 184
- alpha::vector< T > . . . 196

Chapter 4

Class Index

4.1 Class List

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

ApiClient
    Documentation Qt Version 6.9.0 . . . 15
AppDriver . . . 22
Circle
    The Circle class . . . 25
ColumnEditDelegate . . . 34
Ellipse
    The Ellipse class . . . 36
Line
    The Line class . . . 45
LoginWindow . . . 56
MainWindow . . . 59
Parser . . . 70
Polygon
    The Polygon class . . . 76
Polyline . . . 87
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t . . . 97
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t . . . 97
Rectangle . . . 98
RenderArea . . . 106
RenderAreaManager . . . 112
Shape
    The Shape Abstract Base Class . . . 120
ShapesManager . . . 150
Square . . . 156
Testimonial . . . 164
TestimonialDialog . . . 167
TestimonialManager . . . 168
TestimonialsDisplayDialog . . . 171
Text . . . 172
UserAccount . . . 184
UserManager . . . 189
alpha::vector< T > . . . 196

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

src/code/all_shapes.h . . . 205
src/code/ApiClient.cpp . . . 206
src/code/ApiClient.h
Implements the ApiClient class that makes API requests to the Crow Webservice . . . 206
src/code/AppDriver.cpp . . . 209
src/code/AppDriver.h . . . 209
src/code/circle.cpp . . . 211
src/code/circle.h . . . 211
src/code/ColumnEditDelegate.h . . . 212
src/code/ellipse.cpp . . . 214
src/code/ellipse.h . . . 214
src/code/line.cpp . . . 215
src/code/line.h . . . 216
src/code/loginwindow.cpp . . . 217
src/code/loginwindow.h . . . 217
src/code/main.cpp . . . 219
src/code/mainwindow.cpp . . . 220
src/code/mainwindow.h . . . 220
src/code/moc_ApiClient.cpp . . . 223
src/code/moc_UserManager.cpp . . . 224
src/code/Parser.cpp . . . 225
src/code/Parser.h . . . 225
src/code/polygon.cpp . . . 227
src/code/polygon.h . . . 228
src/code/polyline.cpp . . . 229
src/code/polyline.h . . . 229
src/code/rectangle.cpp . . . 231
src/code/rectangle.h . . . 231
src/code/renderarea.cpp . . . 232
src/code/renderarea.h . . . 232
src/code/RenderAreaManager.cpp . . . 234
src/code/RenderAreaManager.h . . . 234
src/code/shape.cpp . . . 236
src/code/shape.h . . . 237
src/code/ShapesManager.cpp . . . 239

src/code/ShapesManager.h . . . 240
src/code/square.cpp . . . 241
src/code/square.h . . . 242
src/code/Testimonial.cpp . . . 243
src/code/Testimonial.h . . . 243
src/code/TestimonialDialog.cpp . . . 244
src/code/TestimonialDialog.h . . . 245
src/code/TestimonialManager.cpp . . . 246
src/code/TestimonialManager.h . . . 246
src/code/TestimonialsDisplayDialog.cpp . . . 248
src/code/TestimonialsDisplayDialog.h . . . 248
src/code/text.cpp . . . 250
src/code/text.h . . . 250
src/code/UserAccount.cpp . . . 252
src/code/UserAccount.h . . . 252
src/code/UserManager.cpp . . . 253
src/code/UserManager.h . . . 254
src/code/vector.h . . . 255
src/code/webservice.cpp
Implements the Crow web service for handling shapes and render area data . . . 260

Chapter 6

Topic Documentation

6.1 > Shapes

Global constant PI used for calculating perimater and area.

Global constant PI used for calculating perimater and area.

Chapter 7

Namespace Documentation

7.1 alpha Namespace Reference

Classes

- class vector

7.2 QT_WARNING_DISABLE_DEPRECATED Namespace Reference

Classes

- struct qt_meta_tag_ZN11UserManagerE_t
- struct qt_meta_tag_ZN9ApiClientE_t

7.3 Ui Namespace Reference

Chapter 8

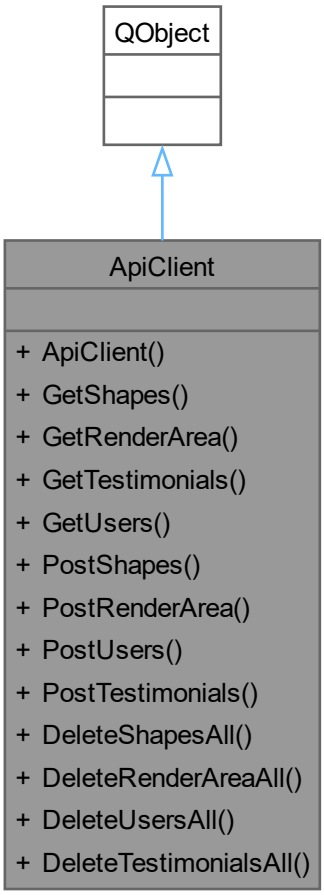
Class Documentation

8.1 ApiClient Class Reference

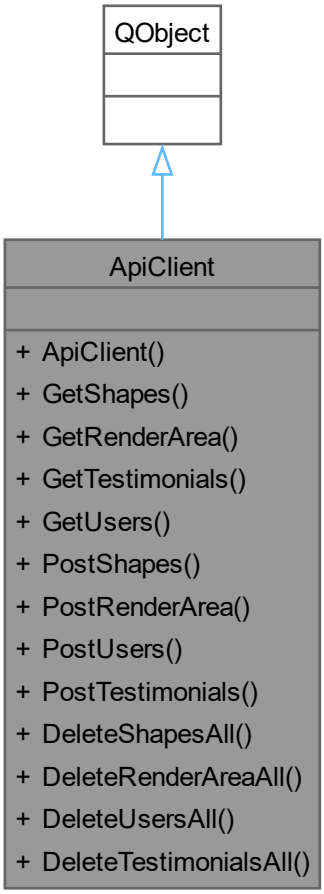
Documentation Qt Version 6.9.0.

`#include <ApiClient.h>`

Inheritance diagram for ApiClient:



Collaboration diagram for ApiClient:



Signals

- void GoodGetReply (const QString &json)

  Signal for a successful request sent to a Get endpoint.
- void BadGetReply (const QString &error)

  Signal for a failed request sent to a Get endpoint.
- void GoodPostReply ()

  Signal for a successful request sent to a Post endpoint.
- void BadPostReply (const QString &error)

  Signal for a failed request sent to a Post endpoint.
- void GoodDeleteReply ()

  Signal for a successful request sent to a Delete endpoint.
- void BadDeleteReply (const QString &error)

  Signal for a failed request sent to a Delete endpoint.

Public Member Functions

- ApiClient (QObject *parent=nullptr)

  Default Constructor.
- void GetShapes ()
- void GetRenderArea ()
- void GetTestimonials ()

  Makes a request to get all testimonials via GET /testimonials.
- void GetUsers ()
- void PostShapes (std::string json)
- void PostRenderArea (std::string json)
- void PostUsers (std::string json)
- void PostTestimonials (std::string json)

  Makes a request to post testimonial data via POST /testimonials.
- void DeleteShapesAll ()
- void DeleteRenderAreaAll ()
- void DeleteUsersAll ()
- void DeleteTestimonialsAll ()

  Makes a request to delete all testimonials via DELETE /testimonials.

8.1.1 Detailed Description

Documentation Qt Version 6.9.0.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 ApiClient()

```
ApiClient::ApiClient (
            QObject * parent = nullptr)  [explicit]
```

Default Constructor.

Parameters

parent	- any QObject to tie this instantiation to ensure automatic deletion

8.1.3 Member Function Documentation

8.1.3.1 BadDeleteReply

```
void ApiClient::BadDeleteReply (
            const QString & error)  [signal]
```

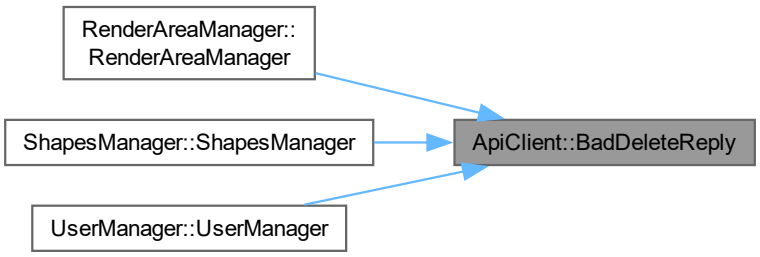
Signal for a failed request sent to a Delete endpoint.

When this signal is emitted, then the API was not able to properly delete the file it needed to. Assume the data sent was not deleted.

Parameters

error	- the error message indicating what went wrong

Here is the caller graph for this function:



8.1.3.2 BadGetReply

```
void ApiClient::BadGetReply (
            const QString & error)  [signal]
```

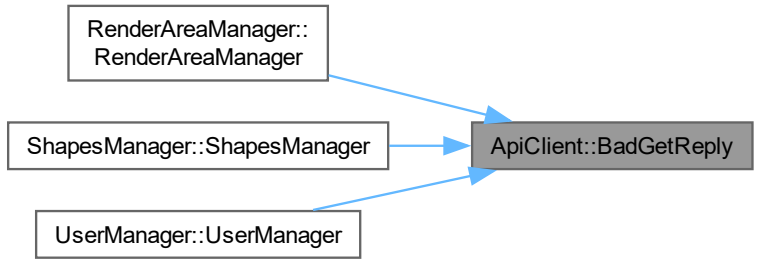
Signal for a failed request sent to a Get endpoint.

When this signal is emitted, the ApiClient object does not have the data we want. Instead, it contains an error message on what went wrong.

Parameters

error	- the error message indicating what went wrong

Here is the caller graph for this function:



8.1.3.3 BadPostReply

```
void ApiClient::BadPostReply (
            const QString & error)  [signal]
```

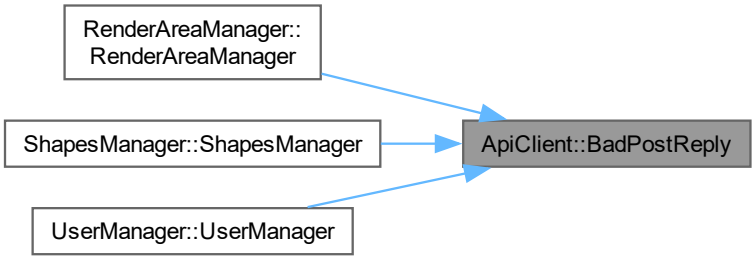
Signal for a failed request sent to a Post endpoint.

When this signal is emitted, then the API was not able to properly update the file it needed to. Assume the data sent was not saved.

Parameters

error	- the error message indicating what went wrong

Here is the caller graph for this function:



8.1.3.4 DeleteRenderAreaAll()

```
void ApiClient::DeleteRenderAreaAll ()
```

8.1.3.5 DeleteShapesAll()

```
void ApiClient::DeleteShapesAll ()
```

8.1.3.6 DeleteTestimonialsAll()

```
void ApiClient::DeleteTestimonialsAll ()
```

Makes a request to delete all testimonials via DELETE /testimonials.

Sends a DELETE to clear the testimonials on the server. Replies are handled by AnalyzeDeleteReply().

8.1.3.7 DeleteUsersAll()

```
void ApiClient::DeleteUsersAll ()
```

8.1.3.8 GetRenderArea()

```
void ApiClient::GetRenderArea ()
```

8.1.3.9 GetShapes()

```
void ApiClient::GetShapes ()
```

8.1.3.10 GetTestimonials()

```
void ApiClient::GetTestimonials ()
```

Makes a request to get all testimonials via GET /testimonials.

Sends a GET to retrieve the JSON array of testimonials. Replies are handled by AnalyzeGetReply().

8.1.3.11 GetUsers()

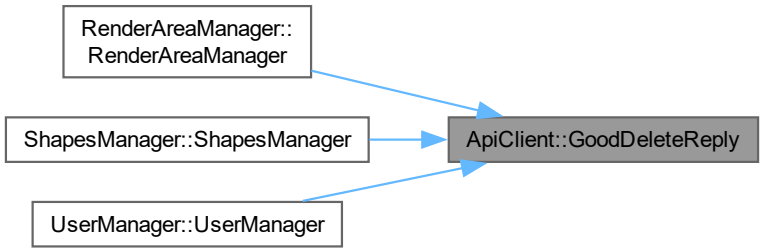
```
void ApiClient::GetUsers ()
```

8.1.3.12 GoodDeleteReply

```
void ApiClient::GoodDeleteReply ()  [signal]
```

Signal for a successful request sent to a Delete endpoint.

When this signal is emitted, then the data was successfully deleted by the API. Here is the caller graph for this function:



8.1.3.13 GoodGetReply

```
void ApiClient::GoodGetReply (
            const QString & json)  [signal]
```

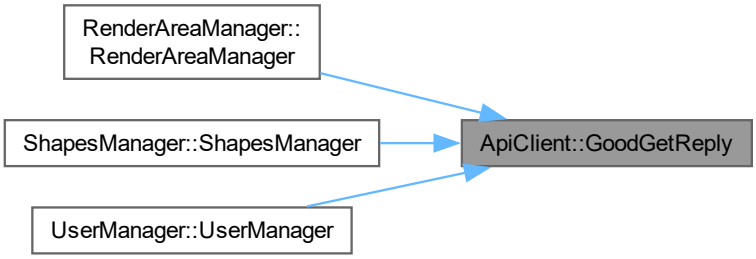
Signal for a successful request sent to a Get endpoint.

When this signal is emitted, the ApiClient object contains the data that we want formatted in JSON.

Parameters

json	- contains data received from whichever Get endpoint was called and is formatted in JSON.

Here is the caller graph for this function:

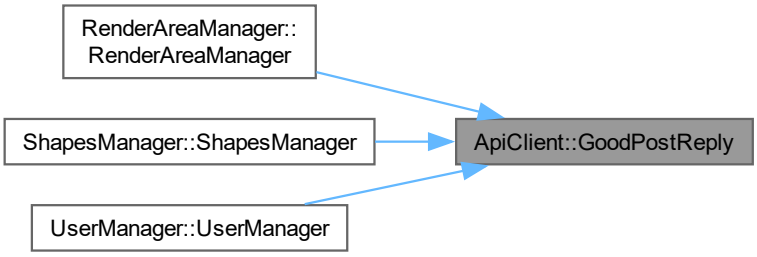


8.1.3.14 GoodPostReply

```
void ApiClient::GoodPostReply ()  [signal]
```

Signal for a successful request sent to a Post endpoint.

When this signal is emitted, then the data was successfully received by the API. Here is the caller graph for this function:



8.1.3.15 PostRenderArea()

```
void ApiClient::PostRenderArea (
            std::string json)
```

8.1.3.16 PostShapes()

```
void ApiClient::PostShapes (
            std::string json)
```

8.1.3.17 PostTestimonials()

```
void ApiClient::PostTestimonials (
            std::string json)
```

Makes a request to post testimonial data via POST /testimonials.

Sends a POST with the JSON string of testimonials. Replies are handled by AnalyzePostReply().

Parameters

json	- JSON-formatted string representing testimonials array.

8.1.3.18 PostUsers()

```
void ApiClient::PostUsers (
            std::string json)
```

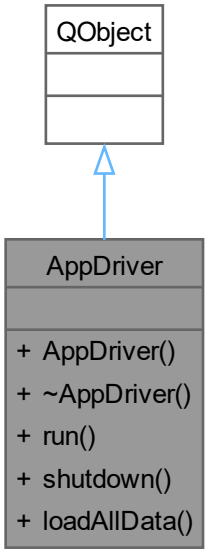
The documentation for this class was generated from the following files:

- src/code/ApiClient.h
- src/code/ApiClient.cpp
- src/code/moc_ApiClient.cpp

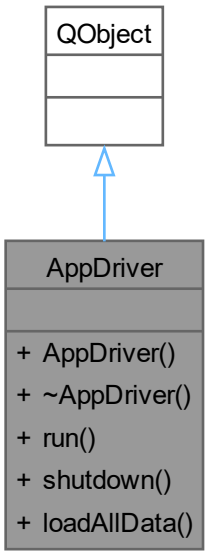
8.2 AppDriver Class Reference

```
#include <AppDriver.h>
```

Inheritance diagram for AppDriver:



Collaboration diagram for AppDriver:



Public Member Functions

- AppDriver (QObject ∗parent=nullptr)

- ~AppDriver ()
- void run ()
- void shutdown ()
- void loadAllData ()

8.2.1 Constructor & Destructor Documentation

8.2.1.1 AppDriver()

```
AppDriver::AppDriver (
            QObject * parent = nullptr)
```

8.2.1.2 ~AppDriver()

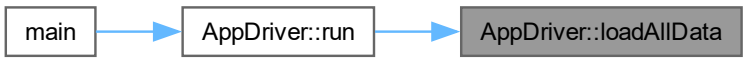
```
AppDriver::~AppDriver ()
```

8.2.2 Member Function Documentation

8.2.2.1 loadAllData()

```
void AppDriver::loadAllData ()
```

Here is the caller graph for this function:



8.2.2.2 run()

```
void AppDriver::run ()
```

Here is the call graph for this function:

Here is the caller graph for this function:

8.2.2.3 shutdown()

```
void AppDriver::shutdown ()
```

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

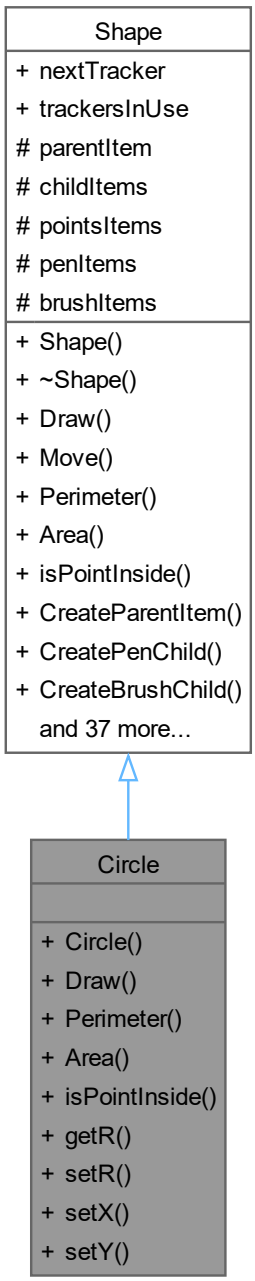
- src/code/AppDriver.h
- src/code/AppDriver.cpp

8.3 Circle Class Reference

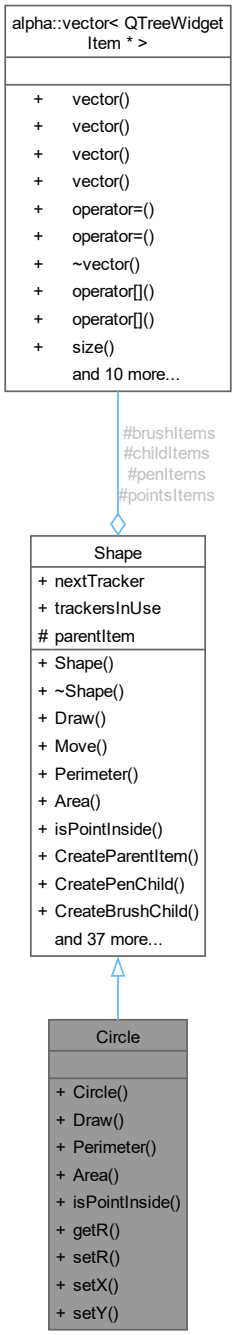
The Circle class.

```
#include "objects/circle.h"
```

Inheritance diagram for Circle:



Collaboration diagram for Circle:



Public Member Functions

- Circle (string shapeType, QPoint coords, QPen pen, QBrush brush, int r)

  Circle Constructor.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws a circle at the assigned coords with the given radius.
- double Perimeter () const override

Perimeter - Returns the perimeter of the circle.

- double Area () const override

  Area - Returns the area of the circle.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns True if point is inside the circle
- int getR () const

  getR - Accessor - Returns the radius
- void setR (int radius)

  Mutator Functions.
- void setX (int x)
- void setY (int y)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- virtual void Move (int x, int y)

  Move - Moves the shape to the x and y coords.
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const

- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem * parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem * > childItems

  vector of QTreeWidgetItem holding data of all child items in parentItem*
- alpha::vector< QTreeWidgetItem * > pointsItems

  vector of QTreeWidgetItem holding data of all points (besides coords) in parentItem*
- alpha::vector< QTreeWidgetItem * > penItems

  vector of QTreeWidgetItem holding data of all pen items*
- alpha::vector< QTreeWidgetItem * > brushItems

  vector of QTreeWidgetItem holding data of all brush items*

8.3.1 Detailed Description

The Circle class.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 Circle()

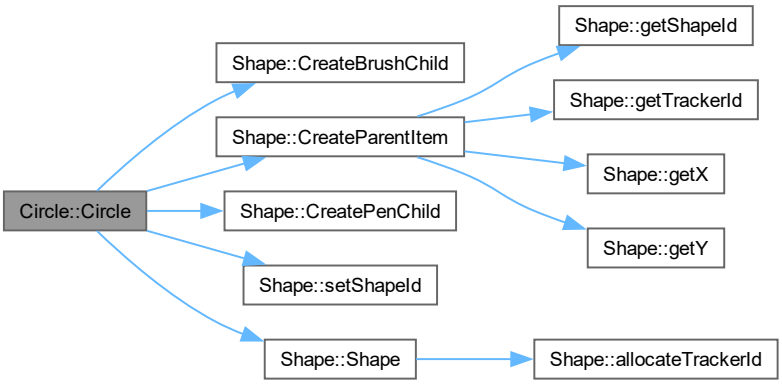
```
Circle::Circle (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            int r)
```

Circle Constructor.

Parameters

shapeType	- Used in Shape constructor MIL
coords	- Used in Shape constructor MIL
pen	- Used in Shape constructor MIL
brush	- Used in Shape constructor MIL
r	- int radius

Here is the call graph for this function:



8.3.3 Member Function Documentation

8.3.3.1 Area()

```
double Circle::Area () const  [override], [virtual]
```

Area - Returns the area of the circle.

Returns

Implements Shape.

8.3.3.2 Draw()

```
void Circle::Draw (
            QWidget * renderArea)  [override], [virtual]
```

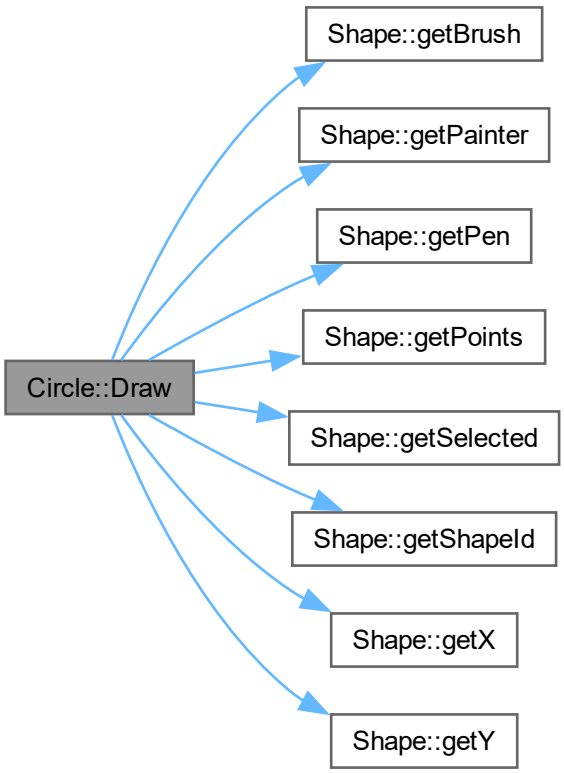
Draw - Draws a circle at the assigned coords with the given radius.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

Here is the call graph for this function:



8.3.3.3 getR()

```
int Circle::getR () const
```

getR - Accessor - Returns the radius

Returns

8.3.3.4 isPointInside()

```
bool Circle::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns True if point is inside the circle

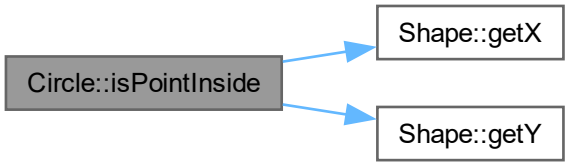
Parameters

point	- point being read

Returns

Implements Shape.

Here is the call graph for this function:



8.3.3.5 Perimeter()

```
double Circle::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the circle.

Returns

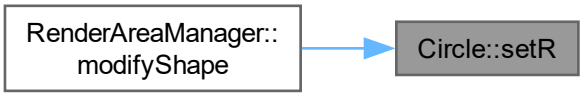
Implements Shape.

8.3.3.6 setR()

```
void Circle::setR (
            int radius)
```

Mutator Functions.

Here is the caller graph for this function:

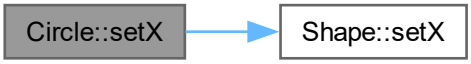


8.3.3.7 setX()

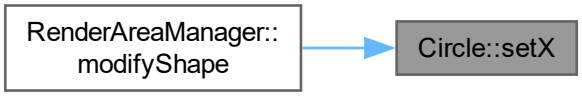
```
void Circle::setX (
            int x)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:

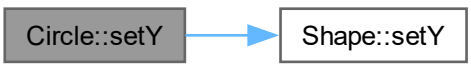


8.3.3.8 setY()

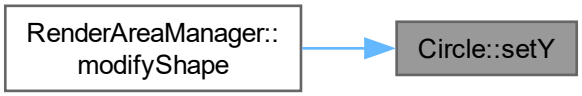
```
void Circle::setY (
            int y)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



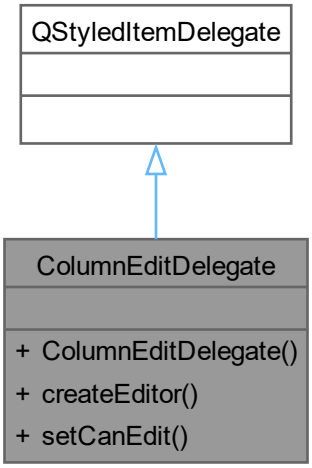
The documentation for this class was generated from the following files:

- src/code/circle.h
- src/code/circle.cpp

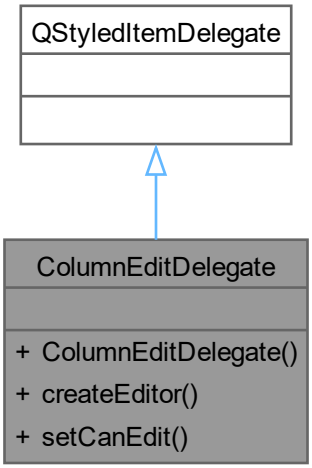
8.4 ColumnEditDelegate Class Reference

`#include <ColumnEditDelegate.h>`

Inheritance diagram for ColumnEditDelegate:



Collaboration diagram for ColumnEditDelegate:



Public Member Functions

- ColumnEditDelegate (QObject ∗parent=nullptr)
- QWidget ∗ createEditor (QWidget ∗parent, const QStyleOptionViewItem &option, const QModelIndex &index) const override
- void setCanEdit (bool edit)

8.4.1 Constructor & Destructor Documentation

8.4.1.1 ColumnEditDelegate()

```
ColumnEditDelegate::ColumnEditDelegate (
            QObject * parent = nullptr)  [inline]
```

8.4.2 Member Function Documentation

8.4.2.1 createEditor()

```
QWidget * ColumnEditDelegate::createEditor (
            QWidget * parent,
            const QStyleOptionViewItem & option,
            const QModelIndex & index) const  [inline], [override]
```

8.4.2.2 setCanEdit()

```
void ColumnEditDelegate::setCanEdit (
            bool edit)  [inline]
```

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

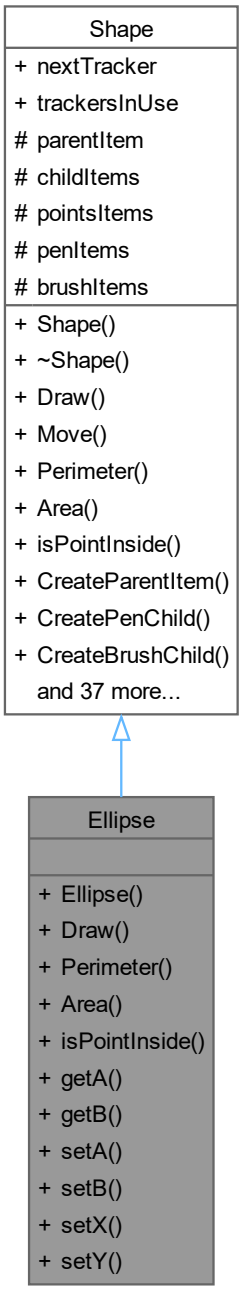
- src/code/ColumnEditDelegate.h

8.5 Ellipse Class Reference

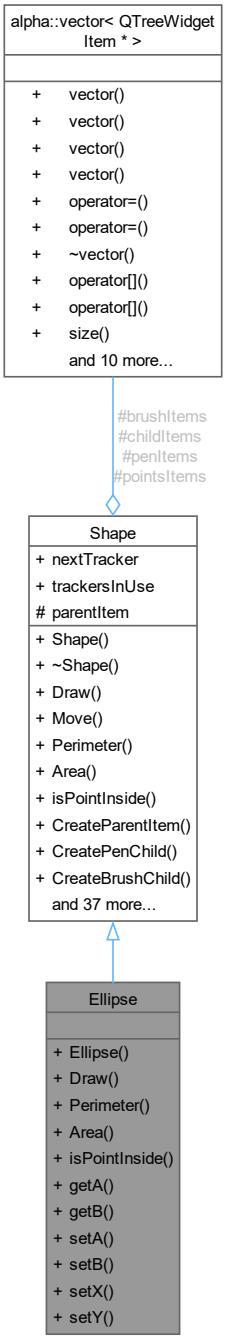
The Ellipse class.

```
#include "objects/ellipse.h"
```

Inheritance diagram for Ellipse:



Collaboration diagram for Ellipse:



Public Member Functions

- Ellipse (string shapeType, QPoint coords, QPen pen, QBrush brush, int a, int b)

  Ellipse Constructor.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws the ellipse to the passed renderArea.
- double Perimeter () const override

Perimeter - Returns the perimeter of the ellipse.

- double Area () const override

  Area - Returns the area of the ellipse.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns True if point is inside the circle
- int getA () const

  Accessor Functions.
- int getB () const
- void setA (int newA)

  Accessor Functions.
- void setB (int newB)
- void setX (int newX)
- void setY (int newY)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- virtual void Move (int x, int y)

  Move - Moves the shape to the x and y coords.
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const

- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem * parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem * > childItems

  vector of QTreeWidgetItem holding data of all child items in parentItem*
- alpha::vector< QTreeWidgetItem * > pointsItems

  vector of QTreeWidgetItem holding data of all points (besides coords) in parentItem*
- alpha::vector< QTreeWidgetItem * > penItems

  vector of QTreeWidgetItem holding data of all pen items*
- alpha::vector< QTreeWidgetItem * > brushItems

  vector of QTreeWidgetItem holding data of all brush items*

8.5.1 Detailed Description

The Ellipse class.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 Ellipse()

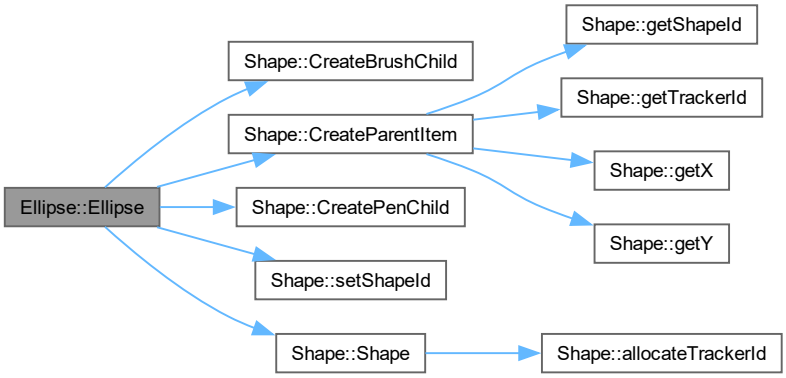
```
Ellipse::Ellipse (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            int a,
            int b)
```

Ellipse Constructor.

Parameters

shapeType	- Used in Shape constructor MIL
coords	- Used in Shape constructor MIL
pen	- Used in Shape constructor MIL
brush	- Used in Shape constructor MIL
a	- The Semi-Minor axis of the Ellipse
b	- The Semi-Major axis of the Ellipse

Here is the call graph for this function:



8.5.3 Member Function Documentation

8.5.3.1 Area()

```
double Ellipse::Area () const  [override], [virtual]
```

Area - Returns the area of the ellipse.

Returns

Implements Shape.

8.5.3.2 Draw()

```
void Ellipse::Draw (
            QWidget * renderArea)  [override], [virtual]
```

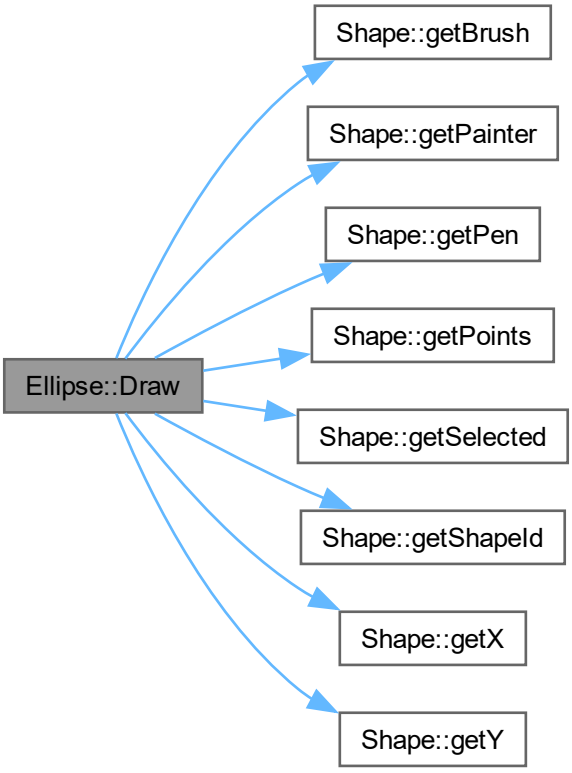
Draw - Draws the ellipse to the passed renderArea.

Parameters

renderArea	- The renderArea which the ellipse is drawn to

Implements Shape.

Here is the call graph for this function:



8.5.3.3 getA()

```
int Ellipse::getA () const
```

Accessor Functions.

8.5.3.4 getB()

```
int Ellipse::getB () const
```

8.5.3.5 isPointInside()

```
bool Ellipse::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns True if point is inside the circle

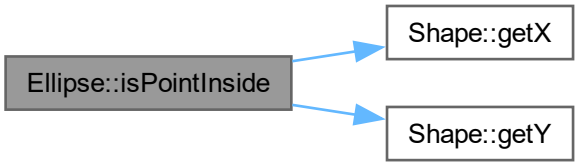
Parameters

point	- point being read

Returns

Implements Shape.

Here is the call graph for this function:



8.5.3.6 Perimeter()

```
double Ellipse::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the ellipse.

Returns

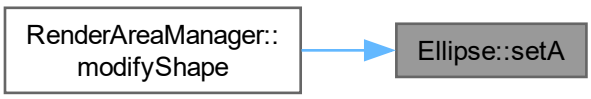
Implements Shape.

8.5.3.7 setA()

```
void Ellipse::setA (
            int newA)
```

Accessor Functions.

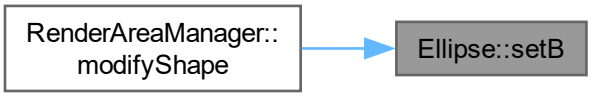
Mutator Functions Here is the caller graph for this function:



8.5.3.8 setB()

```
void Ellipse::setB (
            int newB)
```

Here is the caller graph for this function:

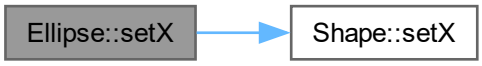


8.5.3.9 setX()

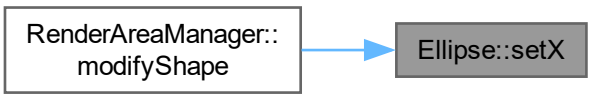
```
void Ellipse::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:

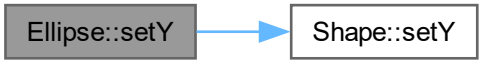


8.5.3.10 setY()

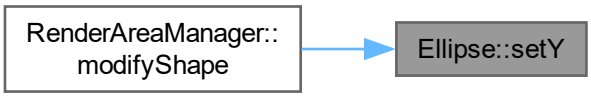
```
void Ellipse::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



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

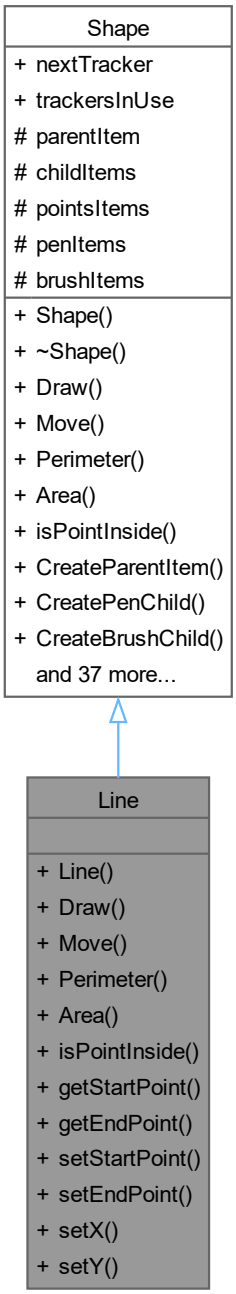
- src/code/ellipse.h
- src/code/ellipse.cpp

8.6 Line Class Reference

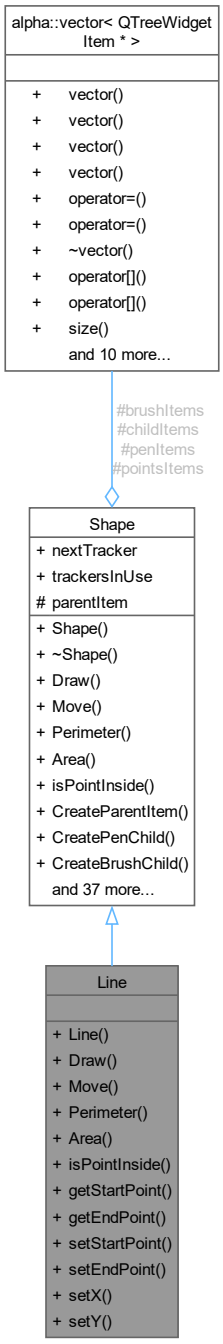
The Line class.

```
#include "objects/line.h"
```

Inheritance diagram for Line:



Collaboration diagram for Line:



Public Member Functions

- Line (string shapeType, QPoint coords, QPen pen, QBrush brush, QPoint startPoint, QPoint endPoint)

  Line Constructor.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws a line from startPoint to endPoint.
- void Move (int x, int y) override

Move - Moves the Line coords to the passed x and y.

- double Perimeter () const override

  Perimeter - Returns the perimeter of the Line.
- double Area () const override

  Area - Returns 0, necessary override to avoid being seen as Abstract.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns True if point is inside the Line
- QPoint getStartPoint () const

  Accessor Functions.
- QPoint getEndPoint () const
- void setStartPoint (const QPoint &newStartPoint)

  Accessor Functions.
- void setEndPoint (const QPoint &newEndPoint)
- void setX (int newX)
- void setY (int newY)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const

- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem ∗ parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem ∗ > childItems

  vector of QTreeWidgetItem∗ holding data of all child items in parentItem
- alpha::vector< QTreeWidgetItem ∗ > pointsItems

  vector of QTreeWidgetItem∗ holding data of all points (besides coords) in parentItem
- alpha::vector< QTreeWidgetItem ∗ > penItems

  vector of QTreeWidgetItem∗ holding data of all pen items
- alpha::vector< QTreeWidgetItem ∗ > brushItems

  vector of QTreeWidgetItem∗ holding data of all brush items

8.6.1 Detailed Description

The Line class.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 Line()

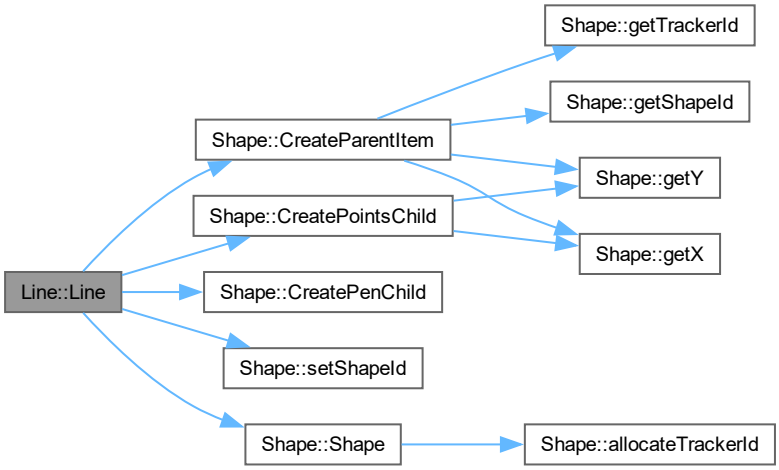
```
Line::Line (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            QPoint startPoint,
            QPoint endPoint)
```

Line Constructor.

Parameters

shapeType	- Used in Shape constructor MIL
coords	- Used in Shape constructor MIL
pen	- Used in Shape constructor MIL
brush	- Used in Shape constructor MIL
startPoint	- Point representing the start of the line
endPoint	- Point representing the end of the line

Here is the call graph for this function:



8.6.3 Member Function Documentation

8.6.3.1 Area()

```
double Line::Area () const  [inline], [override], [virtual]
```

Area - Returns 0, necessary override to avoid being seen as Abstract.

Returns

Implements Shape.

8.6.3.2 Draw()

```
void Line::Draw (
            QWidget * renderArea)  [override], [virtual]
```

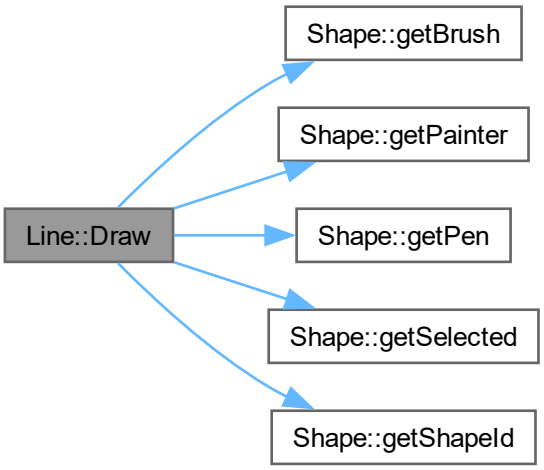
Draw - Draws a line from startPoint to endPoint.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

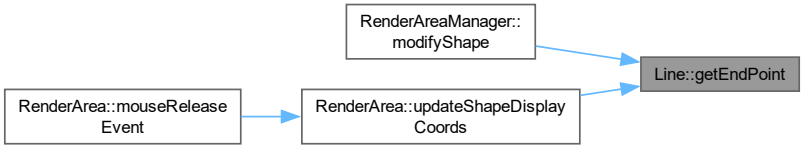
Here is the call graph for this function:



8.6.3.3 getEndPoint()

```
QPoint Line::getEndPoint () const
```

Here is the caller graph for this function:

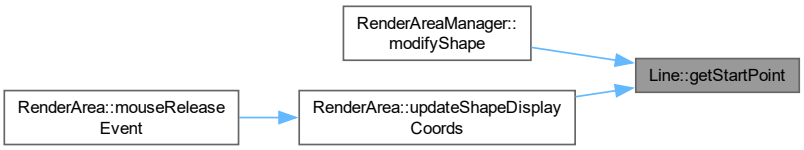


8.6.3.4 getStartPoint()

```
QPoint Line::getStartPoint () const
```

Accessor Functions.

Here is the caller graph for this function:



8.6.3.5 isPointInside()

```
bool Line::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns True if point is inside the Line

Parameters

point	- point being read

Returns

Implements Shape.

8.6.3.6 Move()

```
void Line::Move (
            int x,
            int y)  [override], [virtual]
```

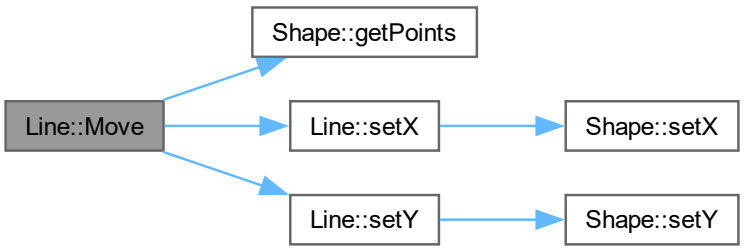
Move - Moves the Line coords to the passed x and y.

Parameters

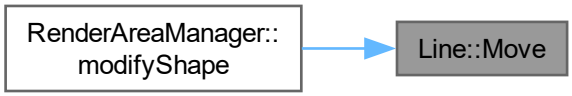
x	- x coordinate
y	- y coordinate

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



8.6.3.7 Perimeter()

```
double Line::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the Line.

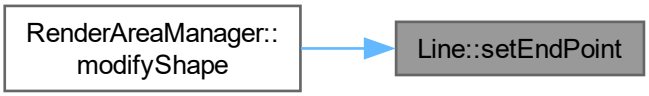
Returns

Implements Shape.

8.6.3.8 setEndPoint()

```
void Line::setEndPoint (
            const QPoint & newEndPoint)
```

Here is the caller graph for this function:

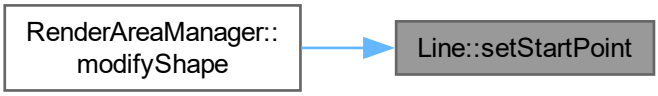


8.6.3.9 setStartPoint()

```
void Line::setStartPoint (
            const QPoint & newStartPoint)
```

Accessor Functions.

Mutator Functions Here is the caller graph for this function:

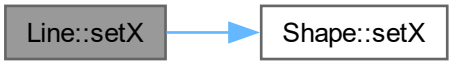


8.6.3.10 setX()

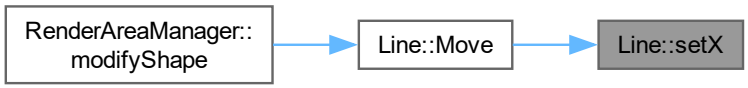
```
void Line::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:

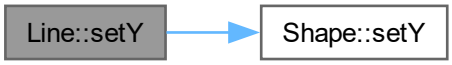


8.6.3.11 setY()

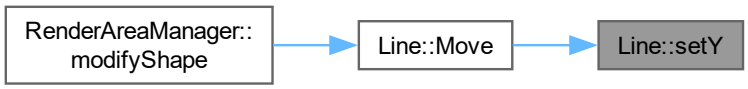
```
void Line::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



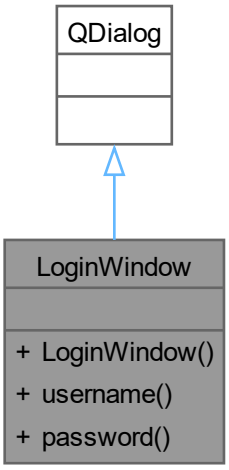
The documentation for this class was generated from the following files:

- src/code/line.h
- src/code/line.cpp

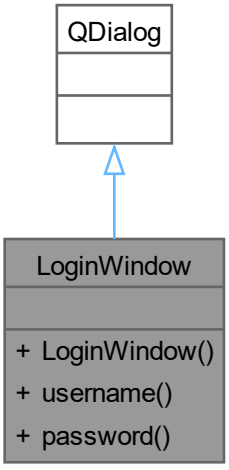
8.7 LoginWindow Class Reference

`#include <loginwindow.h>`

Inheritance diagram for LoginWindow:



Collaboration diagram for LoginWindow:



Signals

- void loginRequested (const QString &username, const QString &password)
- void signupRequested (const QString &username, const QString &password, const bool admin)

Public Member Functions

- LoginWindow (QWidget *parent=nullptr)
- QString username () const
- QString password () const

8.7.1 Constructor & Destructor Documentation

8.7.1.1 LoginWindow()

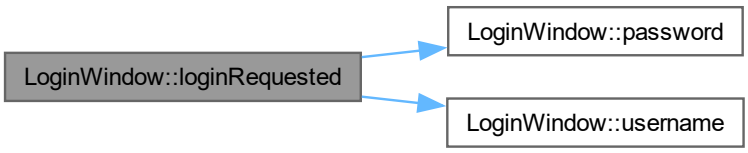
```
LoginWindow::LoginWindow (
            QWidget * parent = nullptr)  [explicit]
```

8.7.2 Member Function Documentation

8.7.2.1 loginRequested

```
void LoginWindow::loginRequested (
            const QString & username,
            const QString & password)  [signal]
```

Here is the call graph for this function:

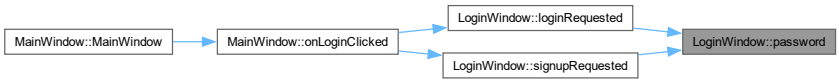


Here is the caller graph for this function:

8.7.2.2 password()

```
QString LoginWindow::password () const
```

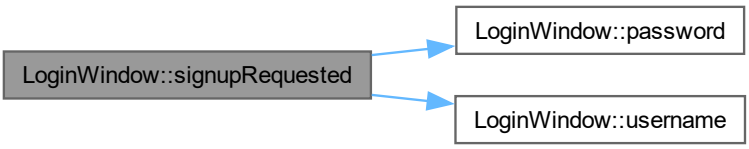
Here is the caller graph for this function:



8.7.2.3 signupRequested

```
void LoginWindow::signupRequested (
            const QString & username,
            const QString & password,
            const bool admin)  [signal]
```

Here is the call graph for this function:

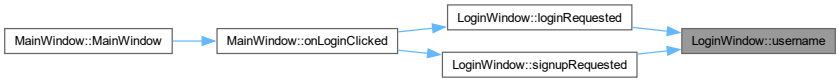


Here is the caller graph for this function:

8.7.2.4 username()

```
QString LoginWindow::username () const
```

Here is the caller graph for this function:



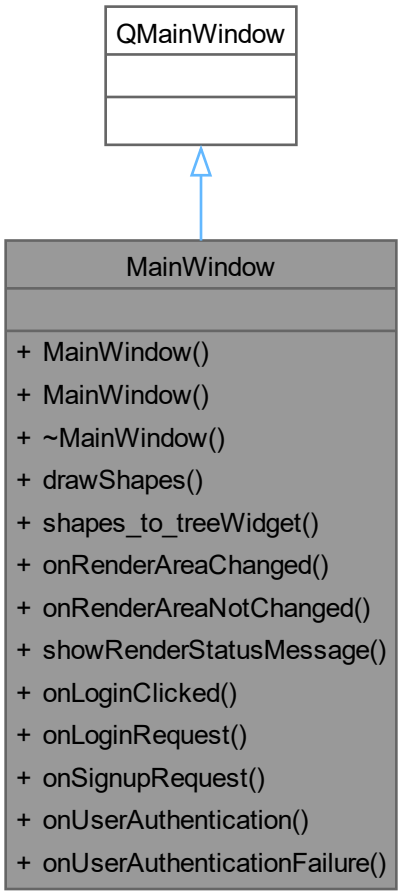
The documentation for this class was generated from the following files:

- src/code/loginwindow.h
- src/code/loginwindow.cpp

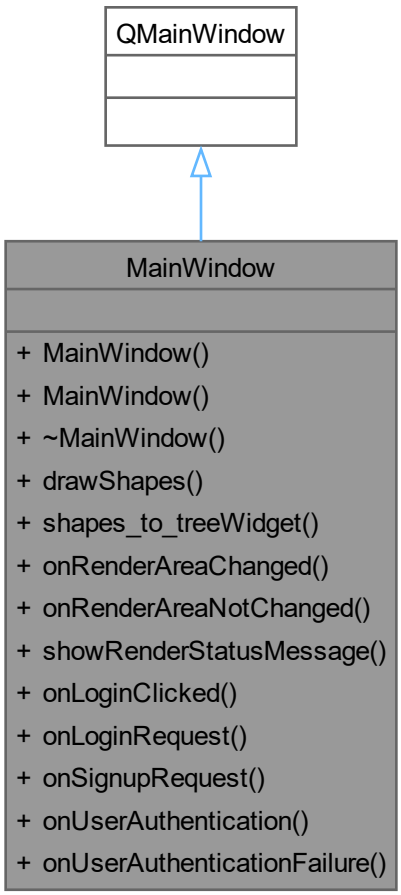
8.8 MainWindow Class Reference

`#include <mainwindow.h>`

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Slots

- void onRenderAreaChanged ()
- void onRenderAreaNotChanged (const QString &message)
- void showRenderStatusMessage (const QString &message)
- void onLoginClicked ()
- void onLoginRequest (const QString &username, const QString &password)
- void onSignupRequest (const QString &username, const QString &password, const bool admin)
- void onUserAuthentication (const UserAccount ∗currUser)
- void onUserAuthenticationFailure (const QString &message)

Signals

- void shapeAdded (Shape ∗shape)
- void shapeChanged (Shape ∗shape, QString key, int vaue)
- void displayedTextChanged (Text ∗text, QString newText)
- void shapeDeleted (int trackerId)

- void deleteAllShapes ()
- void loginAttempt (const QString &username, const QString &password)
- void newUserAdded (const QString &username, const QString &password, const bool admin)
- void loginSuccess ()
- void loginFailed (const QString &message)

Public Member Functions

- MainWindow (QWidget *parent=nullptr)
- MainWindow (QWidget *parent, const alpha::vector< Shape * > *renderedShapes, const UserAccount *currUser)
- ~MainWindow ()
- void drawShapes () const
- void shapes_to_treeWidget ()

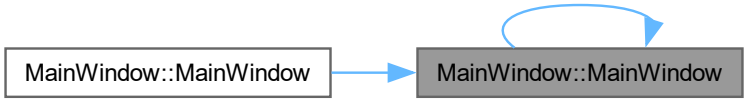
8.8.1 Constructor & Destructor Documentation

8.8.1.1 MainWindow() [1/2]

```
MainWindow::MainWindow (
            QWidget * parent = nullptr)  [explicit]
```

Here is the call graph for this function:

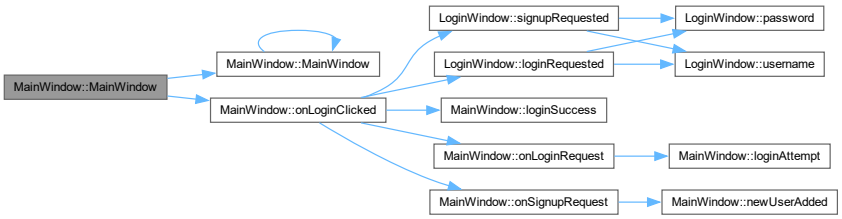
Here is the caller graph for this function:



8.8.1.2 MainWindow() [2/2]

```
MainWindow::MainWindow (
            QWidget * parent,
            const alpha::vector< Shape * > * renderedShapes,
            const UserAccount * currUser)
```

Here is the call graph for this function:



8.8.1.3 ~MainWindow()

```
MainWindow::~MainWindow ()
```

8.8.2 Member Function Documentation

8.8.2.1 deleteAllShapes

```
void MainWindow::deleteAllShapes ()  [signal]
```

8.8.2.2 displayedTextChanged

```
void MainWindow::displayedTextChanged (
            Text * text,
            QString newText)  [signal]
```

8.8.2.3 drawShapes()

```
void MainWindow::drawShapes () const
```

8.8.2.4 loginAttempt

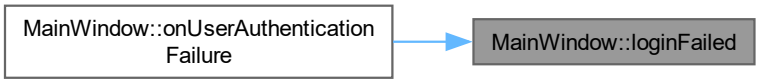
```
void MainWindow::loginAttempt (
            const QString & username,
            const QString & password)  [signal]
```

Here is the caller graph for this function:

8.8.2.5 loginFailed

```
void MainWindow::loginFailed (
            const QString & message)  [signal]
```

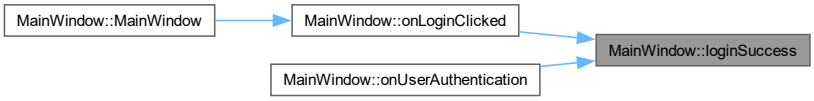
Here is the caller graph for this function:



8.8.2.6 loginSuccess

```
void MainWindow::loginSuccess ()  [signal]
```

Here is the caller graph for this function:



8.8.2.7 newUserAdded

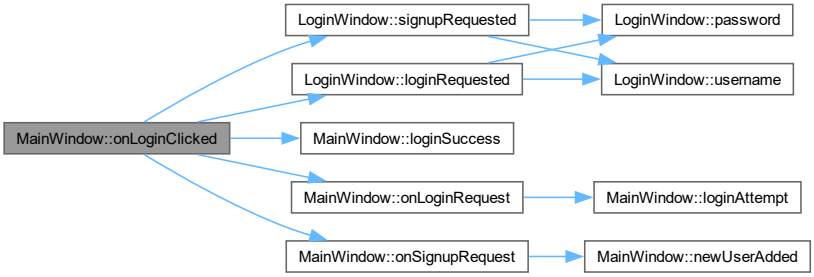
```
void MainWindow::newUserAdded (
            const QString & username,
            const QString & password,
            const bool admin)  [signal]
```

Here is the caller graph for this function:

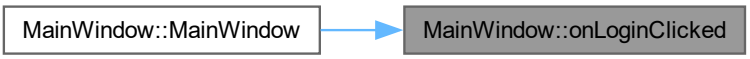
8.8.2.8 onLoginClicked

```
void MainWindow::onLoginClicked ()  [slot]
```

Here is the call graph for this function:



Here is the caller graph for this function:



8.8.2.9 onLoginRequest

```
void MainWindow::onLoginRequest (
            const QString & username,
            const QString & password)  [slot]
```

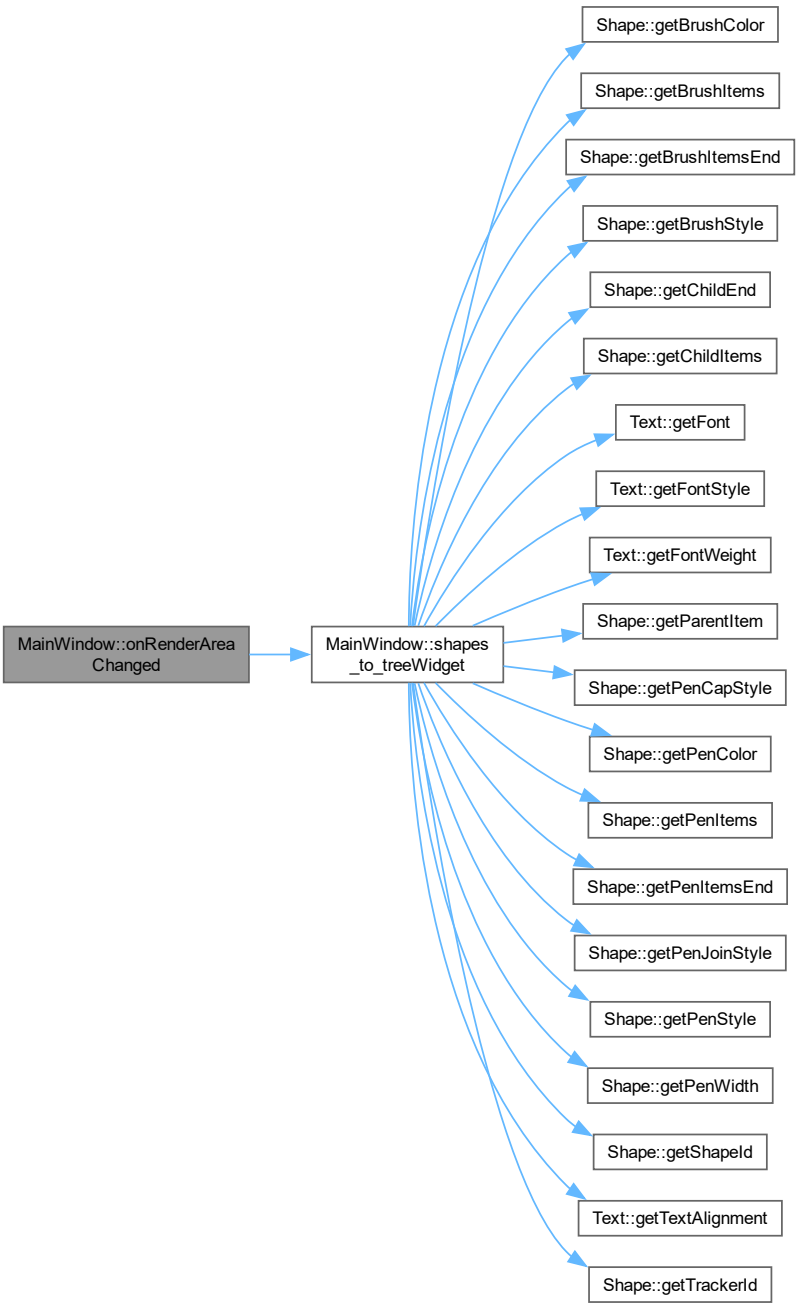
Here is the call graph for this function:

Here is the caller graph for this function:

8.8.2.10 onRenderAreaChanged

```
void MainWindow::onRenderAreaChanged ()  [slot]
```

Here is the call graph for this function:



8.8.2.11 onRenderAreaNotChanged

```
void MainWindow::onRenderAreaNotChanged (
            const QString & message)  [slot]
```

8.8.2.12 onSignupRequest

```
void MainWindow::onSignupRequest (
            const QString & username,
            const QString & password,
            const bool admin)  [slot]
```

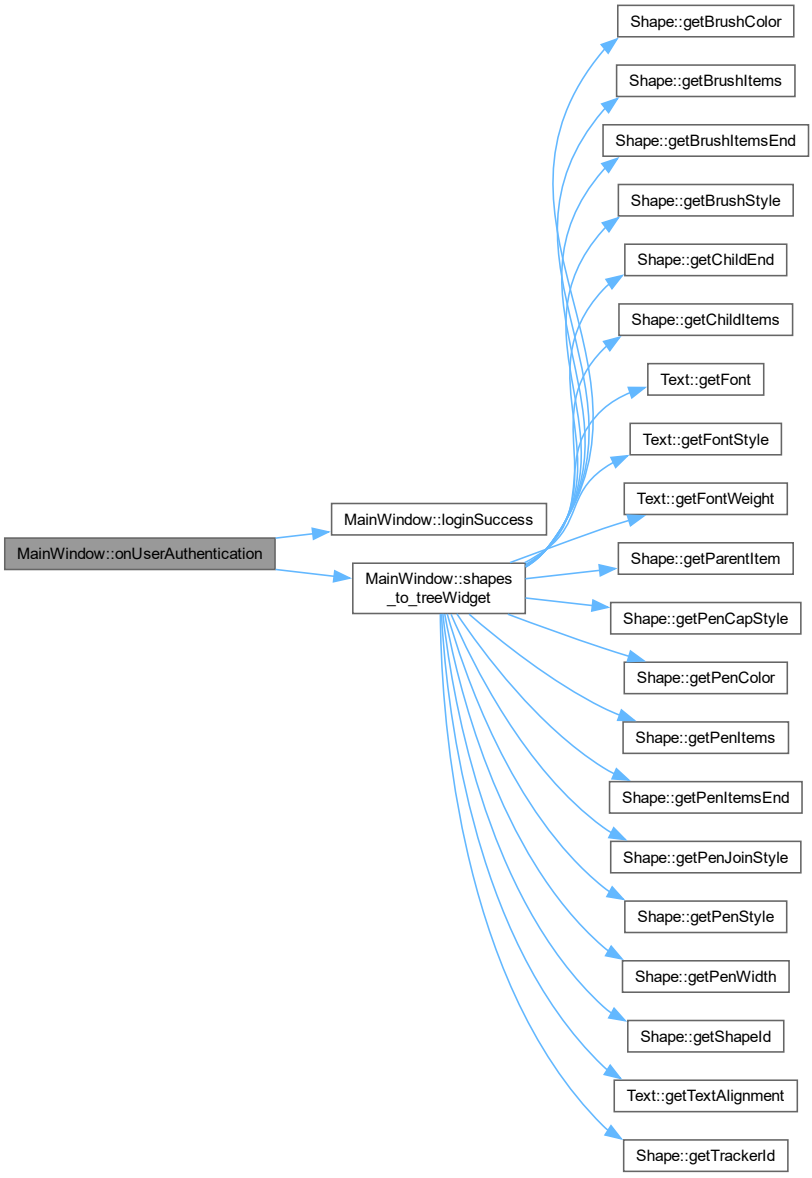
Here is the call graph for this function:

Here is the caller graph for this function:

8.8.2.13 onUserAuthentication

```
void MainWindow::onUserAuthentication (
            const UserAccount * currUser)  [slot]
```

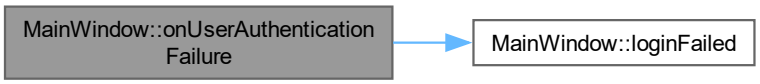
Here is the call graph for this function:



8.8.2.14 onUserAuthenticationFailure

```
void MainWindow::onUserAuthenticationFailure (
            const QString & message)  [slot]
```

Here is the call graph for this function:



8.8.2.15 shapeAdded

```
void MainWindow::shapeAdded (
            Shape * shape)  [signal]
```

8.8.2.16 shapeChanged

```
void MainWindow::shapeChanged (
            Shape * shape,
            QString key,
            int vaue)  [signal]
```

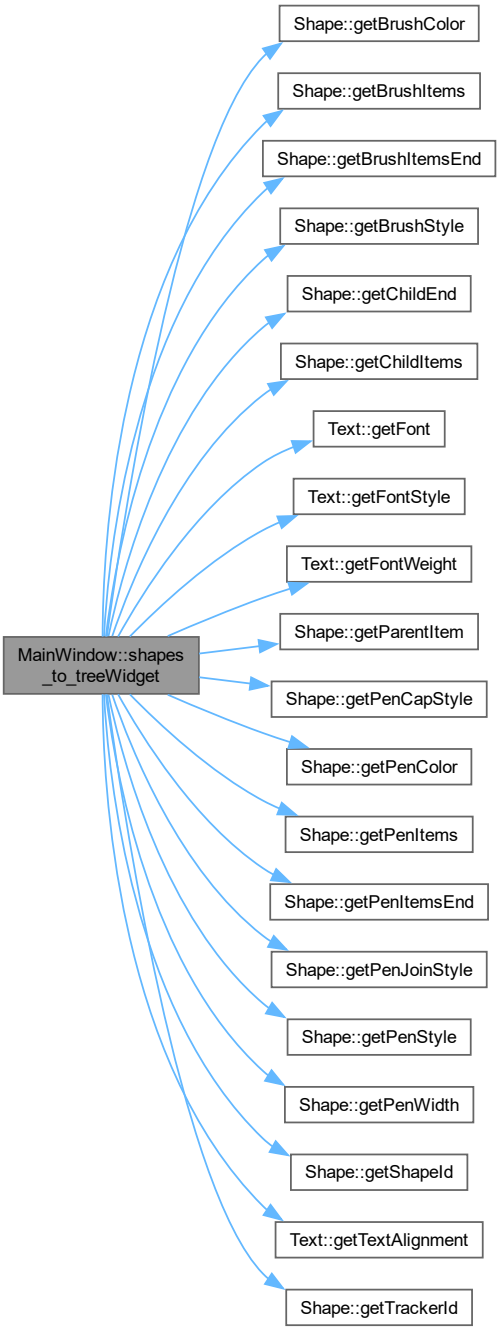
8.8.2.17 shapeDeleted

```
void MainWindow::shapeDeleted (
            int trackerId)  [signal]
```

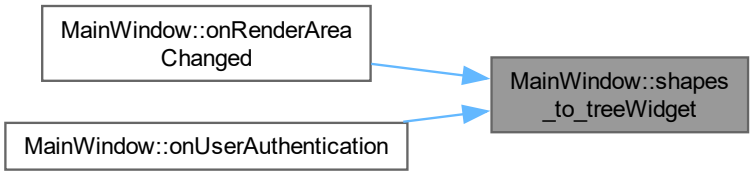
8.8.2.18 shapes_to_treeWidget()

```
void MainWindow::shapes_to_treeWidget ()
```

Here is the call graph for this function:



Here is the caller graph for this function:



8.8.2.19 showRenderStatusMessage

```
void MainWindow::showRenderStatusMessage (
            const QString & message)  [slot]
```

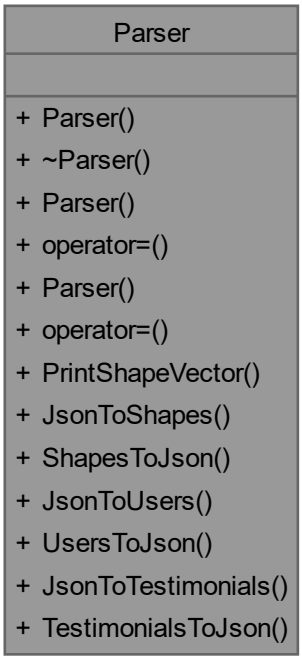
The documentation for this class was generated from the following files:

- src/code/mainwindow.h
- src/code/mainwindow.cpp

8.9 Parser Class Reference

```
#include <Parser.h>
```

Collaboration diagram for Parser:



Public Member Functions

- Parser ()=default
- ~Parser ()=default
- Parser (const Parser &)=delete
- Parser & operator= (const Parser &)=delete
- Parser (Parser &&)=delete
- Parser & operator= (Parser &&)=delete
- void PrintShapeVector (const alpha::vector< Shape * > &shapes)

  This function takes in a vector of Shape pointers and prints some of their primitive data type properties.
- alpha::vector< Shape * > JsonToShapes (const std::string &json)

  This function takes in a vector of Shape pointers and prints some of their primitive data type properties.
- std::string ShapesToJson (const alpha::vector< Shape * > &shapes)

  This function takes in a vector of Shape pointers and converts them into a string formatted in json.
- alpha::vector< UserAccount * > JsonToUsers (const std::string &json)
- std::string UsersToJson (const alpha::vector< UserAccount * > &users)

Static Public Member Functions

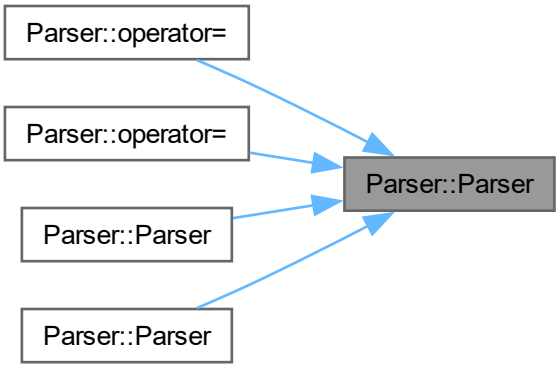
- static QVector< Testimonial > JsonToTestimonials (const std::string &json)
- static std::string TestimonialsToJson (const QVector< Testimonial > &testimonials)

8.9.1 Constructor & Destructor Documentation

8.9.1.1 Parser() [1/3]

```
Parser::Parser ()  [default]
```

Here is the caller graph for this function:



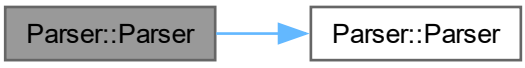
8.9.1.2 ~Parser()

```
Parser::~Parser ()  [default]
```

8.9.1.3 Parser() [2/3]

```
Parser::Parser (
            const Parser & )  [delete]
```

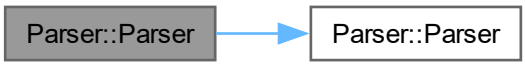
Here is the call graph for this function:



8.9.1.4 Parser() [3/3]

```
Parser::Parser (
            Parser && )  [delete]
```

Here is the call graph for this function:



8.9.2 Member Function Documentation

8.9.2.1 JsonToShapes()

```
alpha::vector< Shape * > Parser::JsonToShapes (
            const std::string & json)
```

This function takes in a vector of Shape pointers and prints some of their primitive data type properties.

Parameters

shapes	- a string formatted in json

Returns

a vector of Shape pointers all pointing to instantiated Shapes

Here is the call graph for this function:

8.9.2.2 JsonToTestimonials()

```
QVector< Testimonial > Parser::JsonToTestimonials (
            const std::string & json)  [static]
```

Here is the call graph for this function:

8.9.2.3 JsonToUsers()

```
alpha::vector< UserAccount * > Parser::JsonToUsers (
            const std::string & json)
```

Here is the call graph for this function:

8.9.2.4 operator=() [1/2]

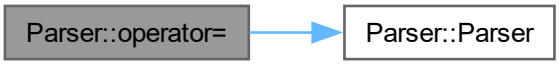
```
Parser & Parser::operator= (
            const Parser & )  [delete]
```

Here is the call graph for this function:

8.9.2.5 operator=() [2/2]

```
Parser & Parser::operator= (
            Parser && )  [delete]
```

Here is the call graph for this function:



8.9.2.6 PrintShapeVector()

```
void Parser::PrintShapeVector (
            const alpha::vector< Shape * > & shapes)
```

This function takes in a vector of Shape pointers and prints some of their primitive data type properties.

Parameters

shapes	- a vector of Shape pointers

8.9.2.7 ShapesToJson()

```
std::string Parser::ShapesToJson (
            const alpha::vector< Shape * > & shapes)
```

This function takes in a vector of Shape pointers and converts them into a string formatted in json.

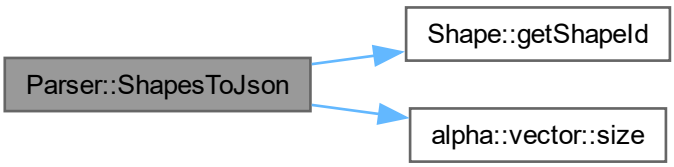
Parameters

shapes	- a vector of Shape pointers

Returns

a string formatted in json containing all of the key:value pairs in the correct json format

Here is the call graph for this function:



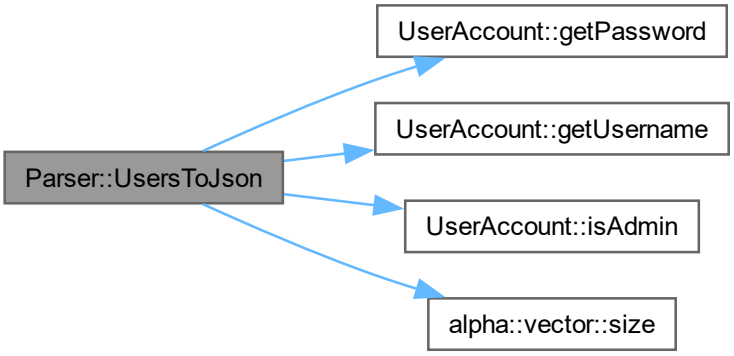
8.9.2.8 TestimonialsToJson()

```
std::string Parser::TestimonialsToJson (
            const QVector< Testimonial > & testimonials)  [static]
```

8.9.2.9 UsersToJson()

```
std::string Parser::UsersToJson (
            const alpha::vector< UserAccount * > & users)
```

Here is the call graph for this function:



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

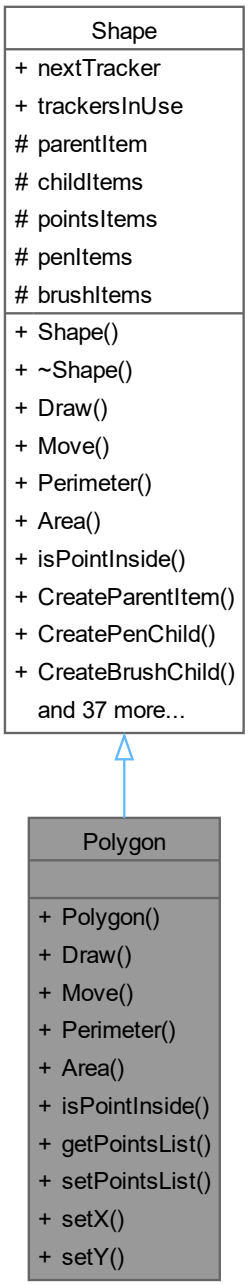
- src/code/Parser.h
- src/code/Parser.cpp

8.10 Polygon Class Reference

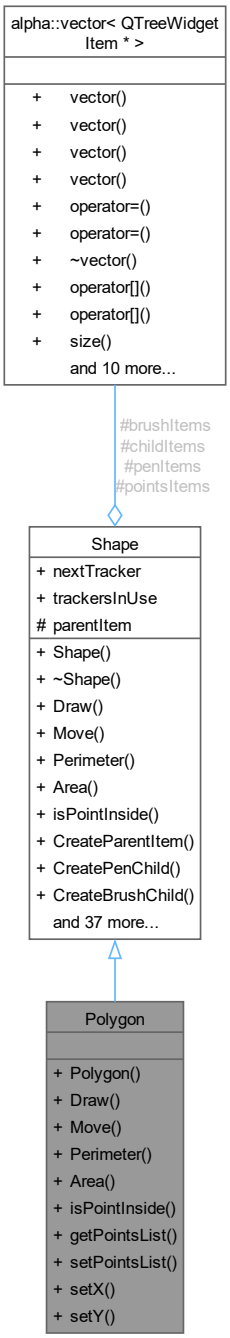
The Polygon class.

```
#include "objects/polygon.h"
```

Inheritance diagram for Polygon:



Collaboration diagram for Polygon:



Public Member Functions

- Polygon (string shapeType, QPoint coords, QPen pen, QBrush brush, QPolygon pointsList)

  Polygon Constructor.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws a Polygon at the assigned coords with points in the pointsList.
- void Move (int x, int y) override

Move - Moves the Polygon to the passed x and y coordinates.

- double Perimeter () const override

  Perimeter - Returns the perimeter of the Polygon.
- double Area () const override

  Area - Returns the area of the Polygon.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns True if point is inside the Polygon
- QPolygon getPointsList () const

  getPointsList - Returns the pointsList by value
- void setPointsList (const QPolygon &newPointsList)

  Mutator Functions.
- void setX (int newX)
- void setY (int newY)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const

- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem ∗ parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem ∗ > childItems

  vector of QTreeWidgetItem∗ holding data of all child items in parentItem
- alpha::vector< QTreeWidgetItem ∗ > pointsItems

  vector of QTreeWidgetItem∗ holding data of all points (besides coords) in parentItem
- alpha::vector< QTreeWidgetItem ∗ > penItems

  vector of QTreeWidgetItem∗ holding data of all pen items
- alpha::vector< QTreeWidgetItem ∗ > brushItems

  vector of QTreeWidgetItem∗ holding data of all brush items

8.10.1 Detailed Description

The Polygon class.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 Polygon()

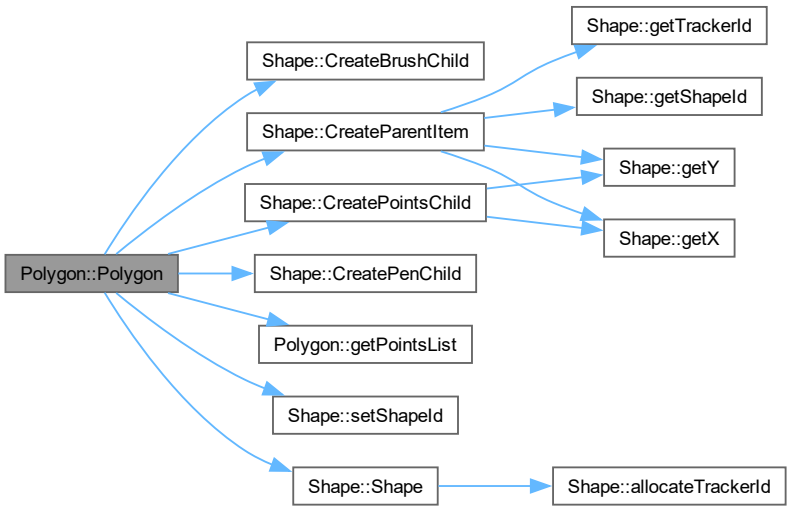
```
Polygon::Polygon (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            QPolygon pointsList)
```

Polygon Constructor.

Parameters

shapeType	- Used in Shape constructor MIL
coords	- Used in Shape constructor MIL
pen	- Used in Shape constructor MIL
brush	- Used in Shape constructor MIL
pointsList	- List of QPoints for each point of the Polygon

Here is the call graph for this function:



8.10.3 Member Function Documentation

8.10.3.1 Area()

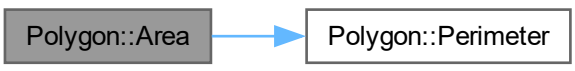
```
double Polygon::Area () const  [override], [virtual]
```

Area - Returns the area of the Polygon.

Returns

Implements Shape.

Here is the call graph for this function:



8.10.3.2 Draw()

```
void Polygon::Draw (
            QWidget * renderArea)  [override], [virtual]
```

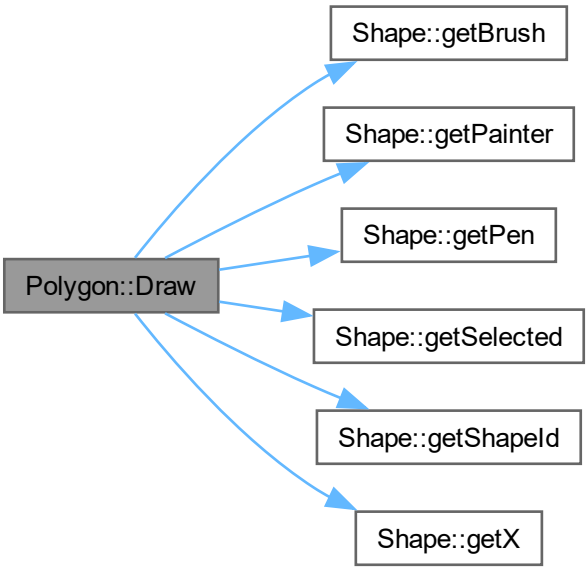
Draw - Draws a Polygon at the assigned coords with points in the pointsList.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

Here is the call graph for this function:



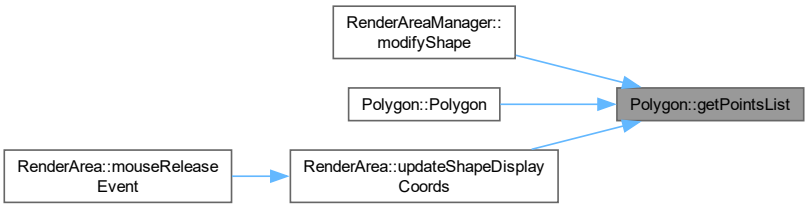
8.10.3.3 getPointsList()

```
QPolygon Polygon::getPointsList () const
```

getPointsList - Returns the pointsList by value

Returns

Here is the caller graph for this function:



8.10.3.4 isPointInside()

```
bool Polygon::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns True if point is inside the Polygon

Parameters

point	- point being read

Returns

Implements Shape.

8.10.3.5 Move()

```
void Polygon::Move (
            int x,
            int y)  [override], [virtual]
```

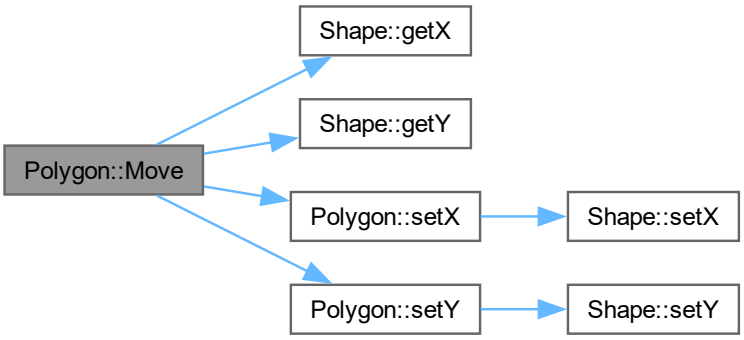
Move - Moves the Polygon to the passed x and y coordinates.

Parameters

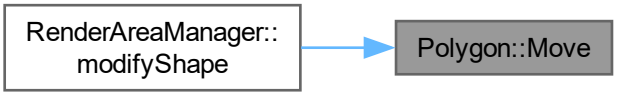
x	- x coordinate
y	- y coordinate

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



8.10.3.6 Perimeter()

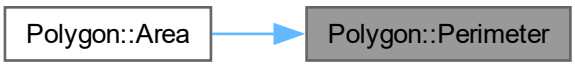
```
double Polygon::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the Polygon.

Returns

Implements Shape.

Here is the caller graph for this function:



8.10.3.7 setPointsList()

```
void Polygon::setPointsList (
            const QPolygon & newPointsList)
```

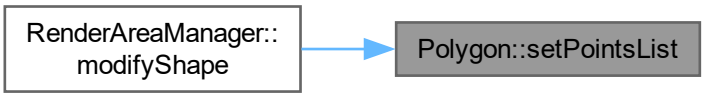
Mutator Functions.

setPointsList - Sets the pointsList to the passed newPointsList

Parameters

newPointsList	

Here is the caller graph for this function:



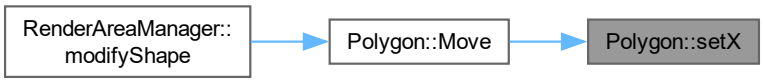
8.10.3.8 setX()

```
void Polygon::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

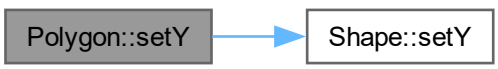


8.10.3.9 setY()

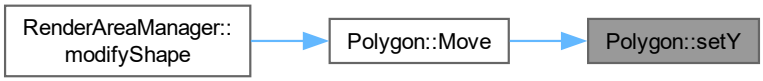
```
void Polygon::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



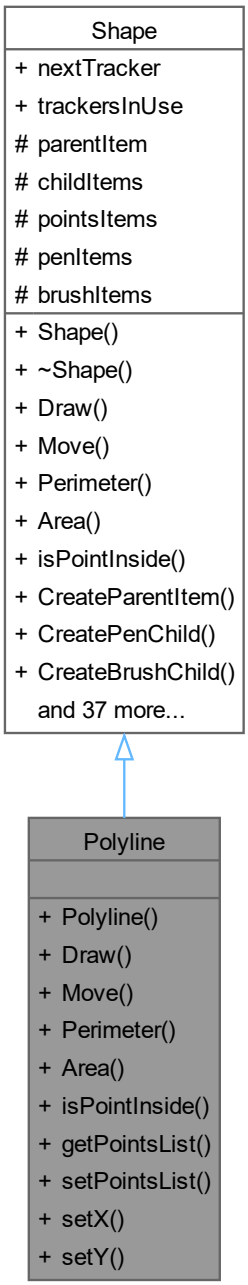
The documentation for this class was generated from the following files:

- src/code/polygon.h
- src/code/polygon.cpp

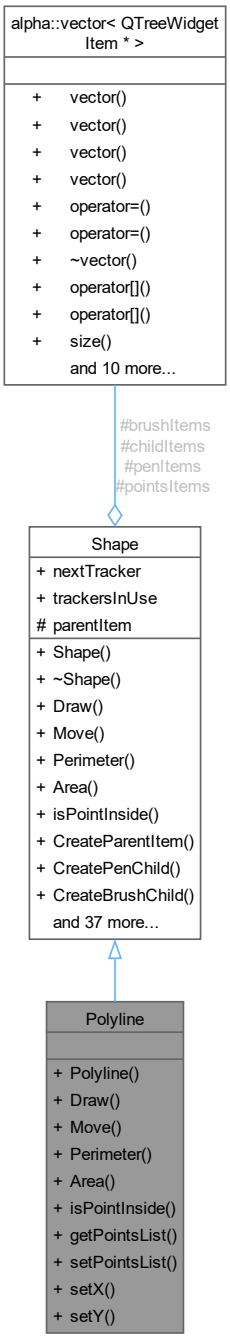
8.11 Polyline Class Reference

```
#include <polyline.h>
```

Inheritance diagram for Polyline:



Collaboration diagram for Polyline:



Public Member Functions

- Polyline (string shapeType, QPoint coords, QPen pen, QBrush brush, QPolygon pointsList)
- void Draw (QWidget ∗renderArea) override

  Draw - Draws the shape to the associated renderArea.
- void Move (int x, int y) override

  Move - Moves the shape to the x and y coords.

- double Perimeter () const override

  Perimeter - Returns the perimeter of the shape.
- double Area () const override

  Area - Returns the area of the shape.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns true if point is inside the shape
- QPolygon getPointsList () const
- void setPointsList (const QPolygon &newPointsList)
- void setX (int newX)
- void setY (int newY)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

Accessor Functions.

- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem ∗ parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem ∗ > childItems

  vector of QTreeWidgetItem∗ holding data of all child items in parentItem
- alpha::vector< QTreeWidgetItem ∗ > pointsItems

  vector of QTreeWidgetItem∗ holding data of all points (besides coords) in parentItem
- alpha::vector< QTreeWidgetItem ∗ > penItems

  vector of QTreeWidgetItem∗ holding data of all pen items
- alpha::vector< QTreeWidgetItem ∗ > brushItems

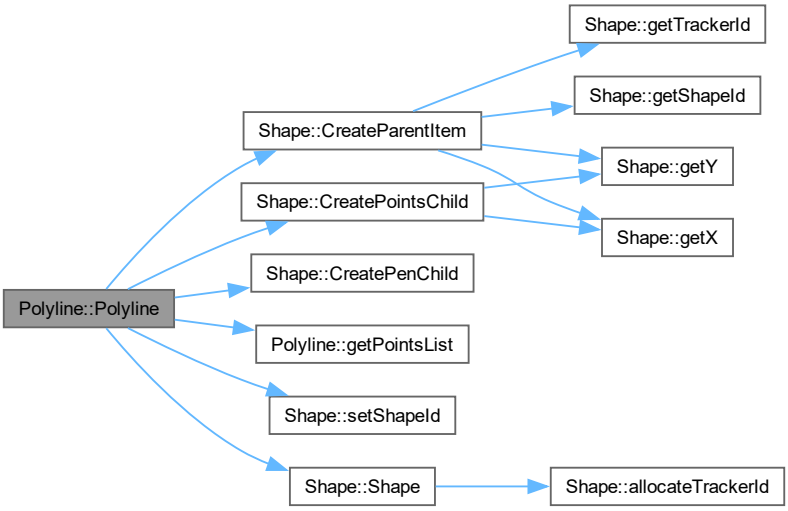
  vector of QTreeWidgetItem∗ holding data of all brush items

8.11.1 Constructor & Destructor Documentation

8.11.1.1 Polyline()

```
Polyline::Polyline (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            QPolygon pointsList)
```

Here is the call graph for this function:



8.11.2 Member Function Documentation

8.11.2.1 Area()

```
double Polyline::Area () const  [inline], [override], [virtual]
```

Area - Returns the area of the shape.

Returns

Implements Shape.

8.11.2.2 Draw()

```
void Polyline::Draw (
            QWidget * renderArea)  [override], [virtual]
```

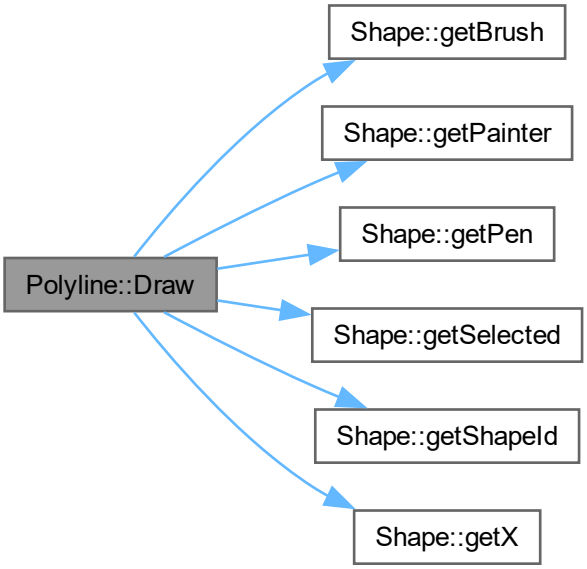
Draw - Draws the shape to the associated renderArea.

Parameters

renderArea	- QWidget to be drawn on

Implements Shape.

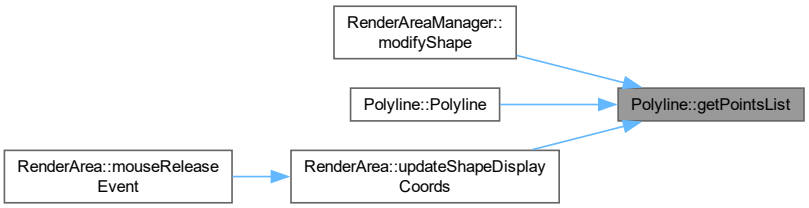
Here is the call graph for this function:



8.11.2.3 getPointsList()

```
QPolygon Polyline::getPointsList () const
```

Here is the caller graph for this function:



8.11.2.4 isPointInside()

```
bool Polyline::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns true if point is inside the shape

Parameters

point	- QPoint being checked

Returns

Implements Shape.

8.11.2.5 Move()

```
void Polyline::Move (
            int x,
            int y)  [override], [virtual]
```

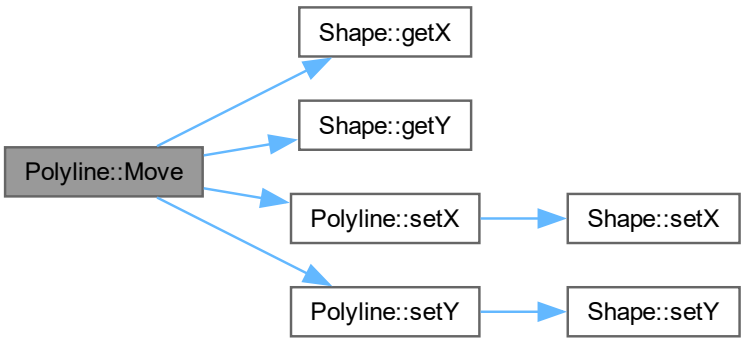
Move - Moves the shape to the x and y coords.

Parameters

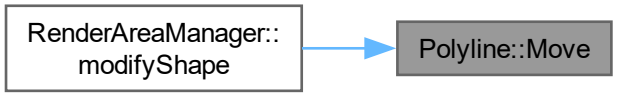
x	- x coordinate
y	- y coordinate

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



8.11.2.6 Perimeter()

```
double Polyline::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the shape.

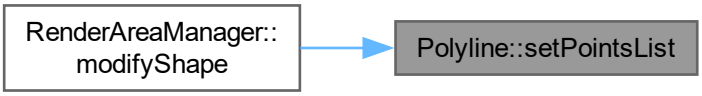
Returns

Implements Shape.

8.11.2.7 setPointsList()

```
void Polyline::setPointsList (
            const QPolygon & newPointsList)
```

Here is the caller graph for this function:



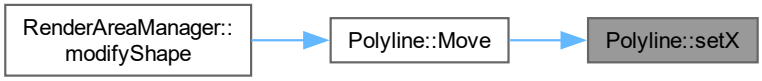
8.11.2.8 setX()

```
void Polyline::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:



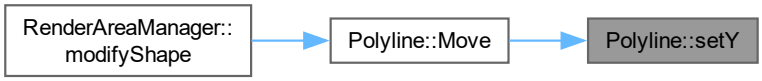
8.11.2.9 setY()

```
void Polyline::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

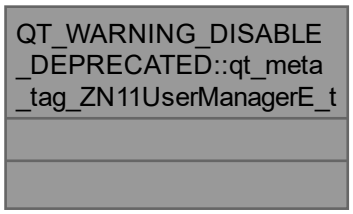


The documentation for this class was generated from the following files:

- src/code/polyline.h
- src/code/polyline.cpp

8.12 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t Struct Reference

Collaboration diagram for QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t:



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

- src/code/moc_UserManager.cpp

8.13 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t Struct Reference

Collaboration diagram for QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t:

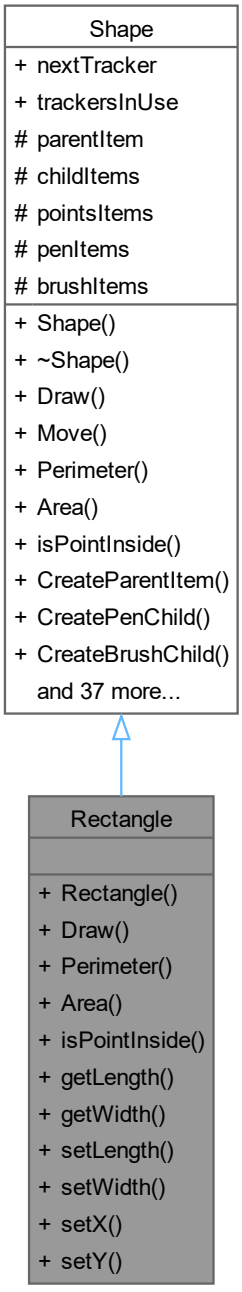
The documentation for this struct was generated from the following file:

- src/code/moc_ApiClient.cpp

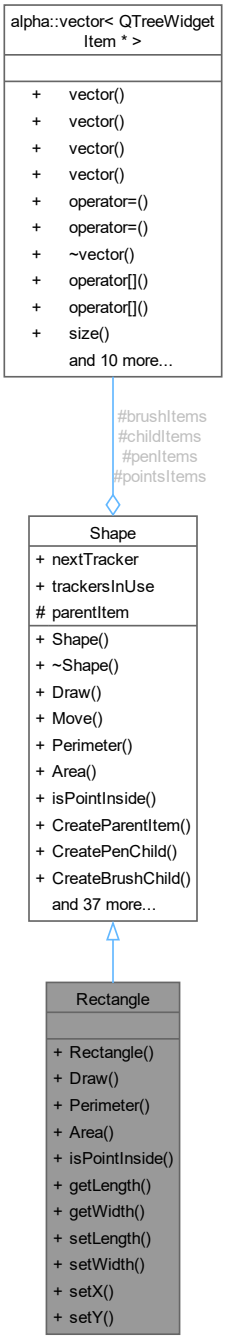
8.14 Rectangle Class Reference

`#include <rectangle.h>`

Inheritance diagram for Rectangle:



Collaboration diagram for Rectangle:



Public Member Functions

- Rectangle (string shapeType, QPoint coords, QPen pen, QBrush brush, int length, int width)
- void Draw (QWidget ∗renderArea) override

  Draw - Draws the shape to the associated renderArea.
- double Perimeter () const override

  Perimeter - Returns the perimeter of the shape.

- double Area () const override

  Area - Returns the area of the shape.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns true if point is inside the shape
- int getLength () const
- int getWidth () const
- void setLength (int newLength)
- void setWidth (int newWidth)
- void setX (int newX)
- void setY (int newY)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- virtual void Move (int x, int y)

  Move - Moves the shape to the x and y coords.
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const

- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem ∗ parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem ∗ > childItems

  vector of QTreeWidgetItem∗ holding data of all child items in parentItem
- alpha::vector< QTreeWidgetItem ∗ > pointsItems

  vector of QTreeWidgetItem∗ holding data of all points (besides coords) in parentItem
- alpha::vector< QTreeWidgetItem ∗ > penItems

  vector of QTreeWidgetItem∗ holding data of all pen items
- alpha::vector< QTreeWidgetItem ∗ > brushItems

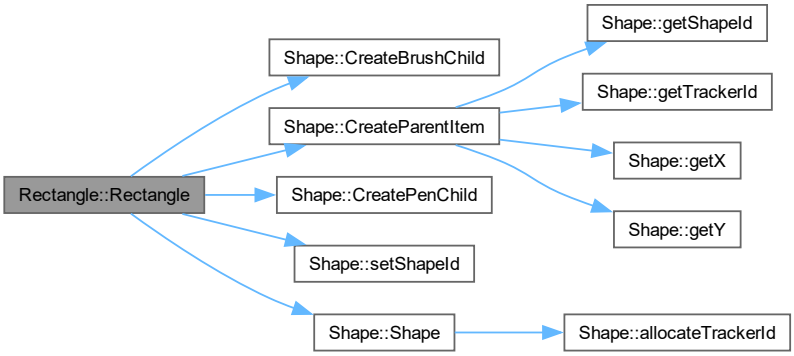
  vector of QTreeWidgetItem∗ holding data of all brush items

8.14.1 Constructor & Destructor Documentation

8.14.1.1 Rectangle()

```
Rectangle::Rectangle (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            int length,
            int width)
```

Here is the call graph for this function:



8.14.2 Member Function Documentation

8.14.2.1 Area()

```
double Rectangle::Area () const  [override], [virtual]
```

Area - Returns the area of the shape.

Returns

Implements Shape.

8.14.2.2 Draw()

```
void Rectangle::Draw (
            QWidget * renderArea)  [override], [virtual]
```

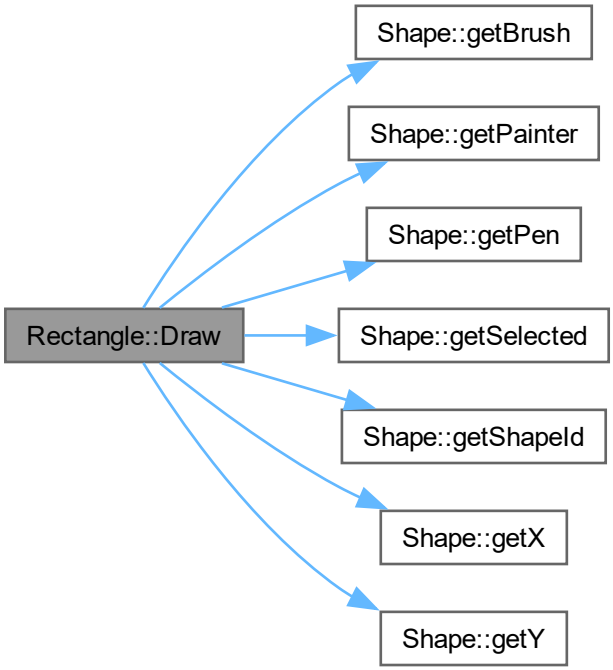
Draw - Draws the shape to the associated renderArea.

Parameters

renderArea	- QWidget to be drawn on

Implements Shape.

Here is the call graph for this function:



8.14.2.3 getLength()

```
int Rectangle::getLength () const
```

8.14.2.4 getWidth()

```
int Rectangle::getWidth () const
```

8.14.2.5 isPointInside()

```
bool Rectangle::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns true if point is inside the shape

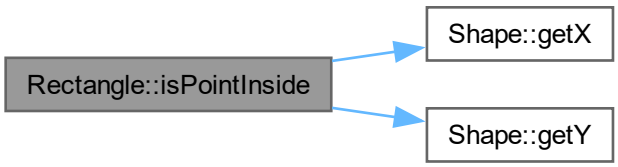
Parameters

point	- QPoint being checked

Returns

Implements Shape.

Here is the call graph for this function:



8.14.2.6 Perimeter()

```
double Rectangle::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the shape.

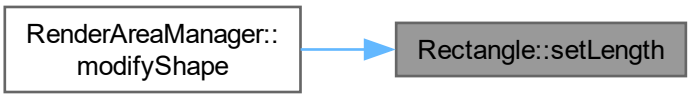
Returns

Implements Shape.

8.14.2.7 setLength()

```
void Rectangle::setLength (
            int newLength)
```

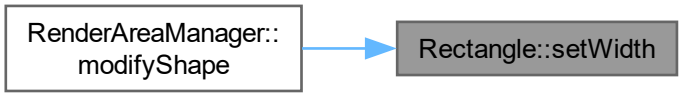
Here is the caller graph for this function:



8.14.2.8 setWidth()

```
void Rectangle::setWidth (
            int newWidth)
```

Here is the caller graph for this function:



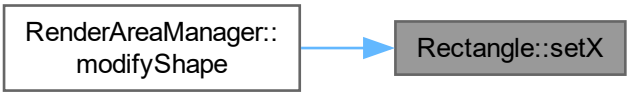
8.14.2.9 setX()

```
void Rectangle::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:



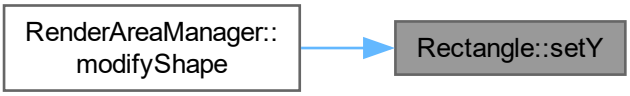
8.14.2.10 setY()

```
void Rectangle::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:



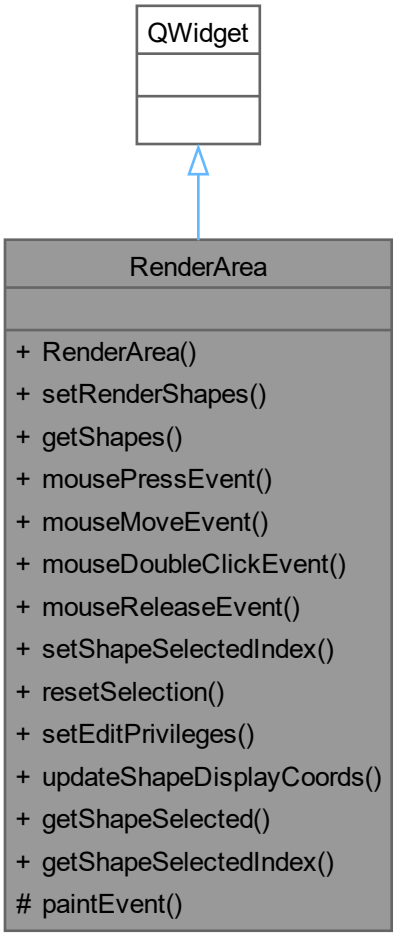
The documentation for this class was generated from the following files:

- src/code/rectangle.h
- src/code/rectangle.cpp

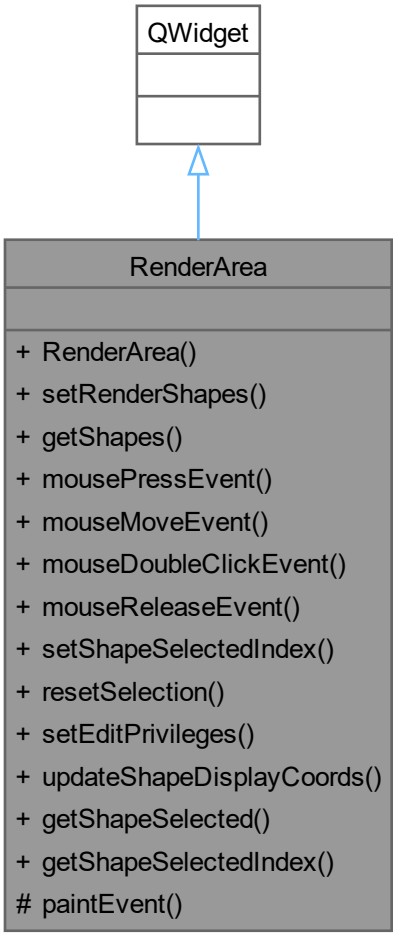
8.15 RenderArea Class Reference

```
#include <renderarea.h>
```

Inheritance diagram for RenderArea:



Collaboration diagram for RenderArea:



Public Member Functions

- RenderArea (QWidget *parent=nullptr)
- void setRenderShapes (const alpha::vector< Shape * > *renderShapes)
- const alpha::vector< Shape * > & getShapes () const
- void mousePressEvent (QMouseEvent *event) override
- void mouseMoveEvent (QMouseEvent *event) override
- void mouseDoubleClickEvent (QMouseEvent *event) override
- void mouseReleaseEvent (QMouseEvent *event) override
- void setShapeSelectedIndex (int newIndex)
- void resetSelection ()
- void setEditPrivileges (bool edit)
- void updateShapeDisplayCoords (Shape *item, const QPoint &position) const
- int getShapeSelected () const
- int getShapeSelectedIndex () const

Protected Member Functions

- void paintEvent (QPaintEvent *event) override

8.15.1 Constructor & Destructor Documentation

8.15.1.1 RenderArea()

```
RenderArea::RenderArea (
            QWidget * parent = nullptr)
```

8.15.2 Member Function Documentation

8.15.2.1 getShapes()

```
const alpha::vector< Shape * > & RenderArea::getShapes () const
```

8.15.2.2 getShapeSelected()

```
int RenderArea::getShapeSelected () const
```

8.15.2.3 getShapeSelectedIndex()

```
int RenderArea::getShapeSelectedIndex () const
```

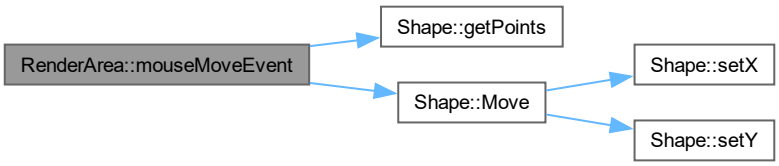
8.15.2.4 mouseDoubleClickEvent()

```
void RenderArea::mouseDoubleClickEvent (
            QMouseEvent * event)  [override]
```

8.15.2.5 mouseMoveEvent()

```
void RenderArea::mouseMoveEvent (
            QMouseEvent * event)  [override]
```

Here is the call graph for this function:



8.15.2.6 mousePressEvent()

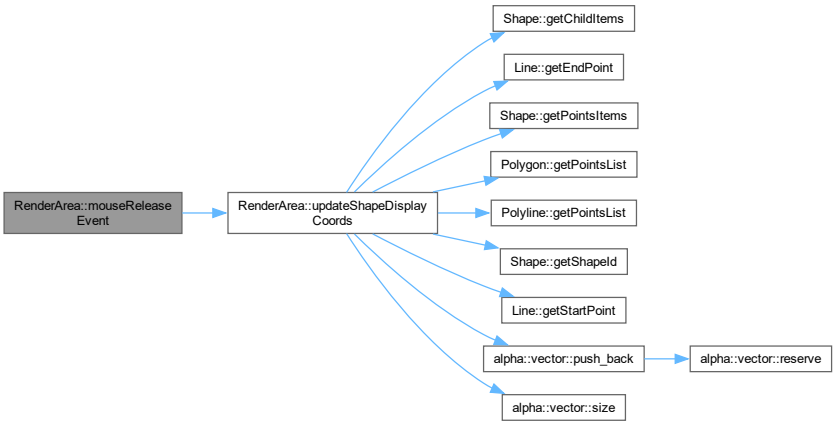
```
void RenderArea::mousePressEvent (
            QMouseEvent * event)  [override]
```

Here is the call graph for this function:

8.15.2.7 mouseReleaseEvent()

```
void RenderArea::mouseReleaseEvent (
            QMouseEvent * event)  [override]
```

Here is the call graph for this function:



8.15.2.8 paintEvent()

```
void RenderArea::paintEvent (
            QPaintEvent * event)  [override], [protected]
```

8.15.2.9 resetSelection()

```
void RenderArea::resetSelection ()
```

8.15.2.10 setEditPrivileges()

```
void RenderArea::setEditPrivileges (
            bool edit)
```

8.15.2.11 setRenderShapes()

```
void RenderArea::setRenderShapes (
            const alpha::vector< Shape * > * renderShapes)
```

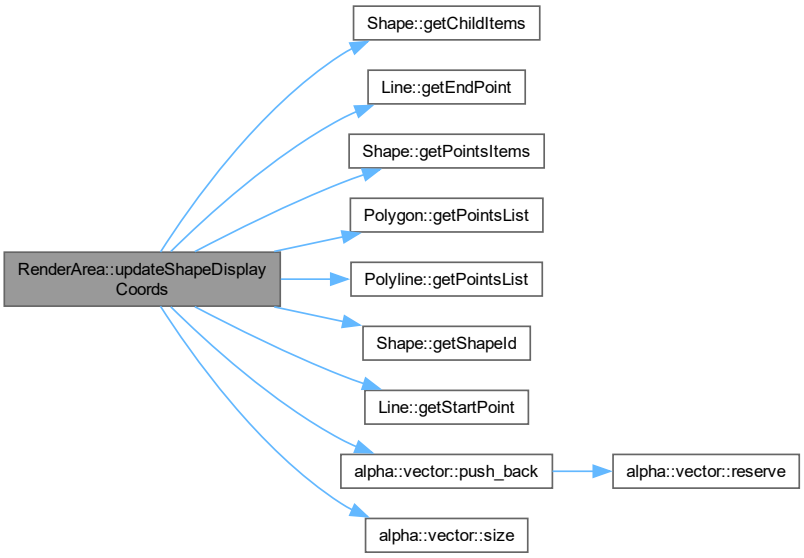
8.15.2.12 setShapeSelectedIndex()

```
void RenderArea::setShapeSelectedIndex (
            int newIndex)
```

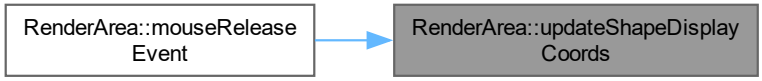
8.15.2.13 updateShapeDisplayCoords()

```
void RenderArea::updateShapeDisplayCoords (
            Shape * item,
            const QPoint & position) const
```

Here is the call graph for this function:



Here is the caller graph for this function:



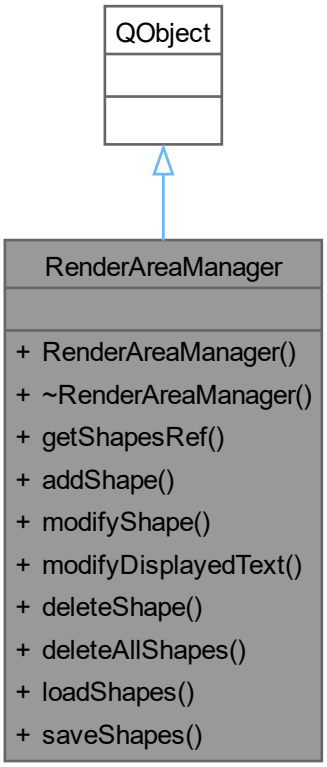
The documentation for this class was generated from the following files:

- src/code/renderarea.h
- src/code/renderarea.cpp

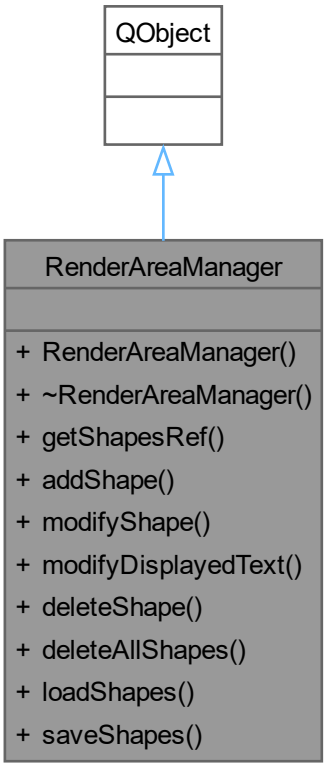
8.16 RenderAreaManager Class Reference

`#include <RenderAreaManager.h>`

Inheritance diagram for RenderAreaManager:



Collaboration diagram for RenderAreaManager:



Signals

- void renderAreaChanged ()
- void renderAreaNotChanged (const QString &message)
- void statusMessage (const QString &message)

Public Member Functions

- RenderAreaManager (QObject ∗parent=nullptr)

  Default Constructor.
- ∼RenderAreaManager ()
- alpha::vector< Shape ∗ > ∗ getShapesRef ()

  Returns the renderShapes vector as a reference.
- void addShape (Shape ∗shape)

  These functions are used to add, change, delete, and load shapes in the RenderAreaManager.
- void modifyShape (Shape ∗shape, QString key, int value)
- void modifyDisplayedText (Text ∗obj, QString newText)
- void deleteShape (const int trackerId)
- void deleteAllShapes ()
- void loadShapes ()
- void saveShapes ()

8.16.1 Constructor & Destructor Documentation

8.16.1.1 RenderAreaManager()

```
RenderAreaManager::RenderAreaManager (
            QObject * parent = nullptr)  [explicit]
```

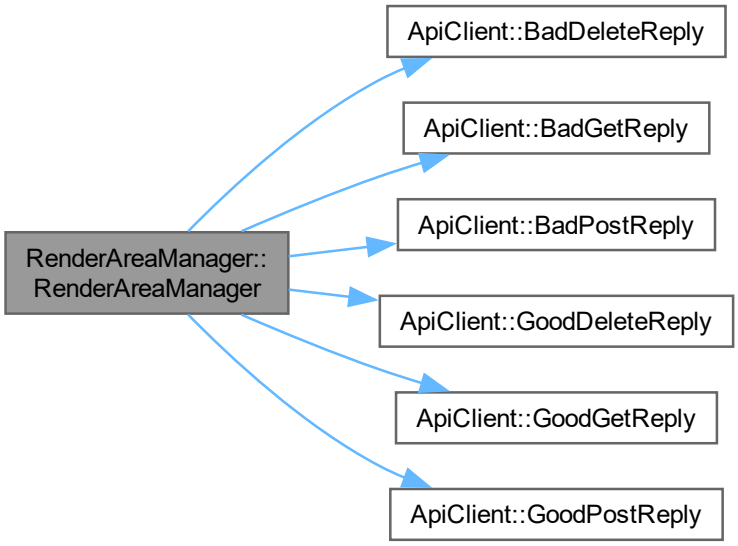
Default Constructor.

This constructor initializes the RenderAreaManager object and connects to the ApiClient signals for handling shape data.

Parameters

parent	- any QObject to tie this instantiation to ensure automatic deletion

Here is the call graph for this function:



8.16.1.2 ~RenderAreaManager()

```
RenderAreaManager::~RenderAreaManager ()
```

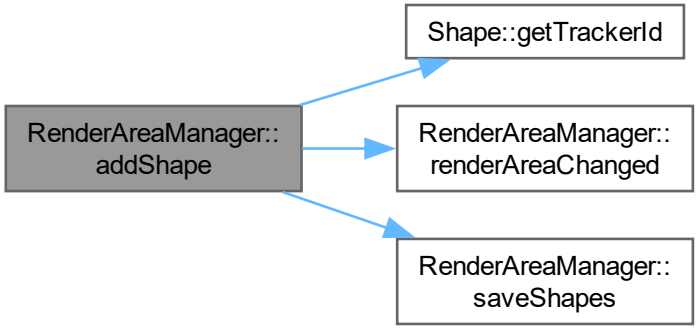
8.16.2 Member Function Documentation

8.16.2.1 addShape()

```
void RenderAreaManager::addShape (
            Shape * shape)
```

These functions are used to add, change, delete, and load shapes in the RenderAreaManager.

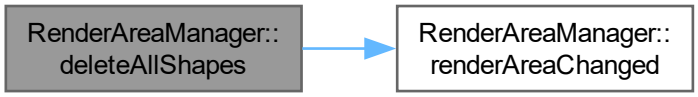
All of these functions emit the renderAreaChanged() signal to notify the frontend that the shapes have changed and need to be redrawn except for saveShapes Here is the call graph for this function:



8.16.2.2 deleteAllShapes()

```
void RenderAreaManager::deleteAllShapes ()
```

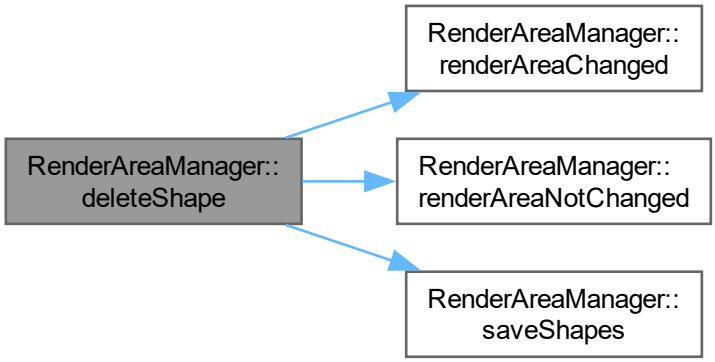
Here is the call graph for this function:



8.16.2.3 deleteShape()

```
void RenderAreaManager::deleteShape (
            const int trackerId)
```

Here is the call graph for this function:



8.16.2.4 getShapesRef()

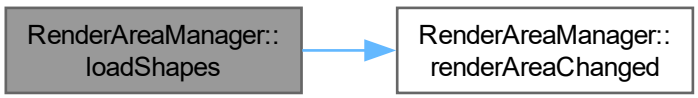
```
alpha::vector< Shape * > * RenderAreaManager::getShapesRef ()
```

Returns the renderShapes vector as a reference.

8.16.2.5 loadShapes()

```
void RenderAreaManager::loadShapes ()
```

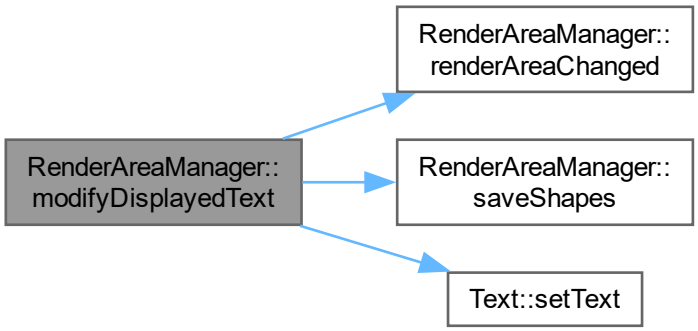
Here is the call graph for this function:



8.16.2.6 modifyDisplayedText()

```
void RenderAreaManager::modifyDisplayedText (
            Text * obj,
            QString newText)
```

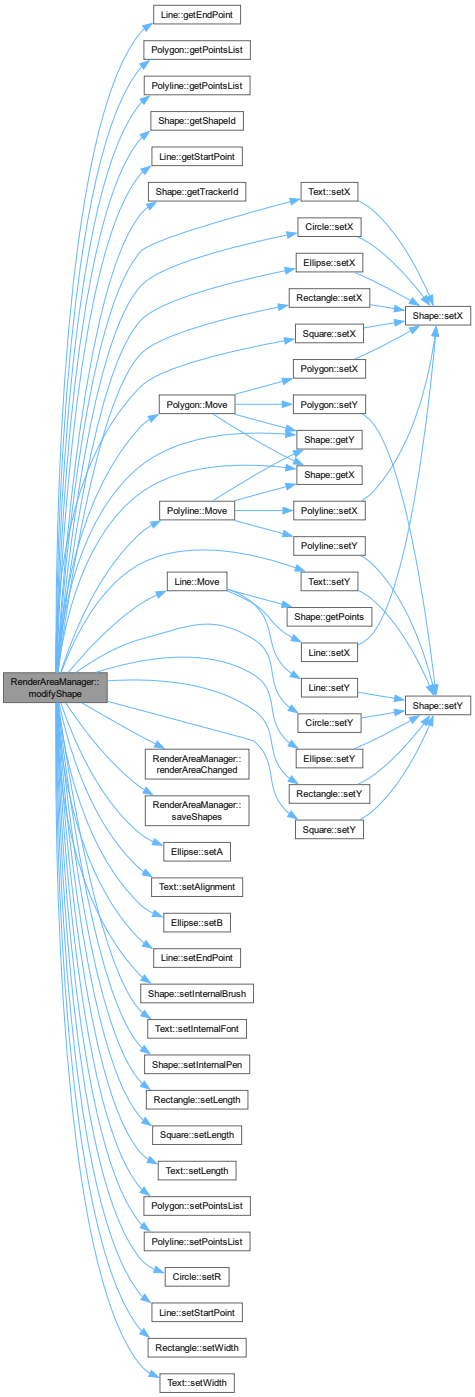
Here is the call graph for this function:



8.16.2.7 modifyShape()

```
void RenderAreaManager::modifyShape (
            Shape * shape,
            QString key,
            int value)
```

Here is the call graph for this function:

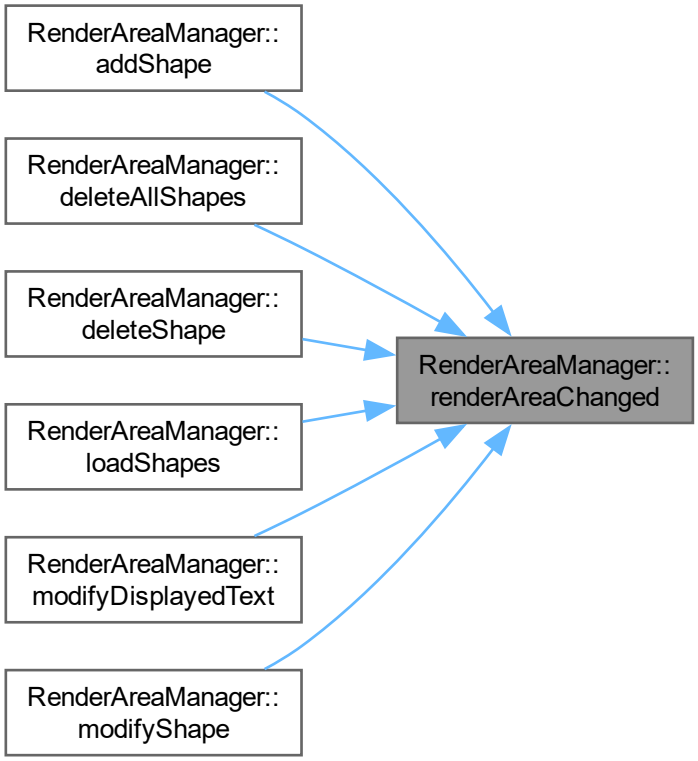


8.16.2.8 renderAreaChanged

```
void RenderAreaManager::renderAreaChanged ()  [signal]
```

These signals are meant to connect to the RenderArea Window in the frontend. -renderAreaChanged(): signal for when the renderShapes vector is changed in anyway so that the frontend knows to refresh the window and redraw

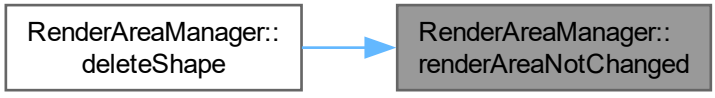
the shapes -statusMessage(): signal for slot functions below. It passes the message to the frontend so it can display a popup saying the shapes were saved successfully, or whatever the message is for the user. Here is the caller graph for this function:



8.16.2.9 renderAreaNotChanged

```
void RenderAreaManager::renderAreaNotChanged (
            const QString & message)  [signal]
```

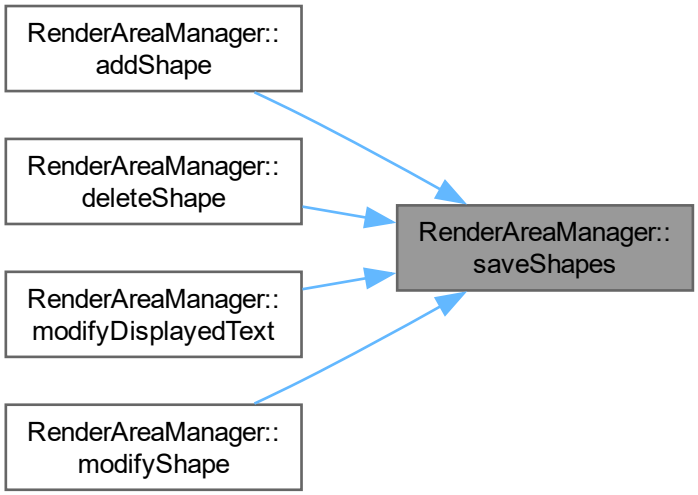
Here is the caller graph for this function:



8.16.2.10 saveShapes()

```
void RenderAreaManager::saveShapes ()
```

Here is the caller graph for this function:



8.16.2.11 statusMessage

```
void RenderAreaManager::statusMessage (
            const QString & message)  [signal]
```

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

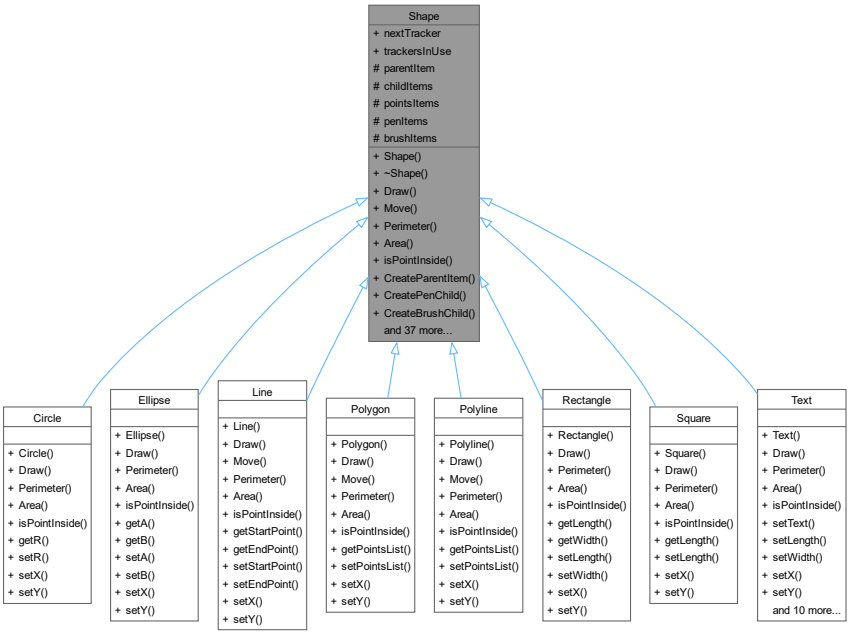
- src/code/RenderAreaManager.h
- src/code/RenderAreaManager.cpp

8.17 Shape Interface Reference

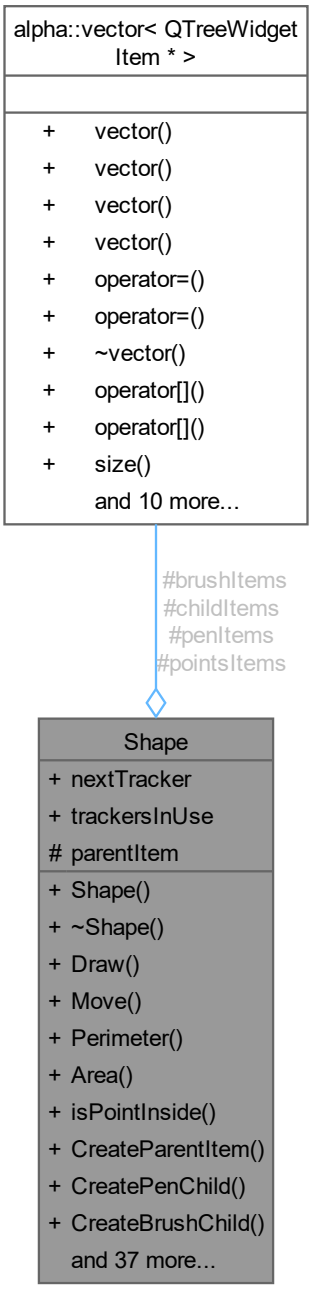
The Shape Abstract Base Class.

```
#include "objects/shape.h"
```

Inheritance diagram for Shape:



Collaboration diagram for Shape:



Public Member Functions

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- virtual void Draw (QWidget *renderArea)=0

Draw - Draws the shape to the associated renderArea.

- virtual void Move (int x, int y)

  Move - Moves the shape to the x and y coords.
- virtual double Perimeter () const =0

  Perimeter - Returns the perimeter of the shape.
- virtual double Area () const =0

  Area - Returns the area of the shape.
- virtual bool isPointInside (const QPoint &point) const =0

  isPointInside - Returns true if point is inside the shape
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setX (int x)
- void setY (int y)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Static Public Attributes

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes

- QTreeWidgetItem * parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem * > childItems

  vector of QTreeWidgetItem holding data of all child items in parentItem*
- alpha::vector< QTreeWidgetItem * > pointsItems

  vector of QTreeWidgetItem holding data of all points (besides coords) in parentItem*
- alpha::vector< QTreeWidgetItem * > penItems

  vector of QTreeWidgetItem holding data of all pen items*
- alpha::vector< QTreeWidgetItem * > brushItems

  vector of QTreeWidgetItem holding data of all brush items*

Friends

- bool operator== (const Shape &shape1, const Shape &shape2)

  operator == - Overloaded equality operator for comparing two shapeId's
- bool operator< (const Shape &shape1, const Shape &shape2)

  operator < - Overloaded less than operator for comparing two shapeId's

8.17.1 Detailed Description

The Shape Abstract Base Class.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 Shape()

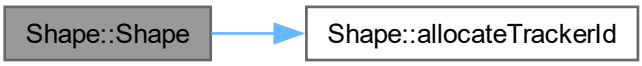
```
Shape::Shape (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush)
```

Shape Constructor.

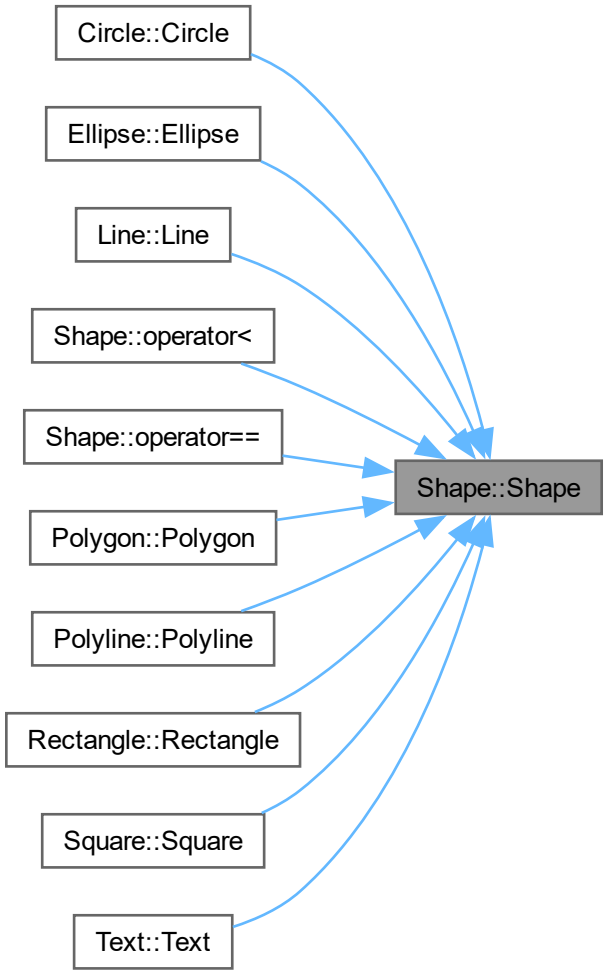
Parameters

shapeType	- string representing shape type, (Line, Circle, etc)
coords	- QPoint with coordinates of the shape
pen	- QPen for the outline of the shape
brush	- QBrush for the fill of the shape

Here is the call graph for this function:



Here is the caller graph for this function:



8.17.2.2 ~Shape()

```
Shape::~Shape ()  [virtual]
```

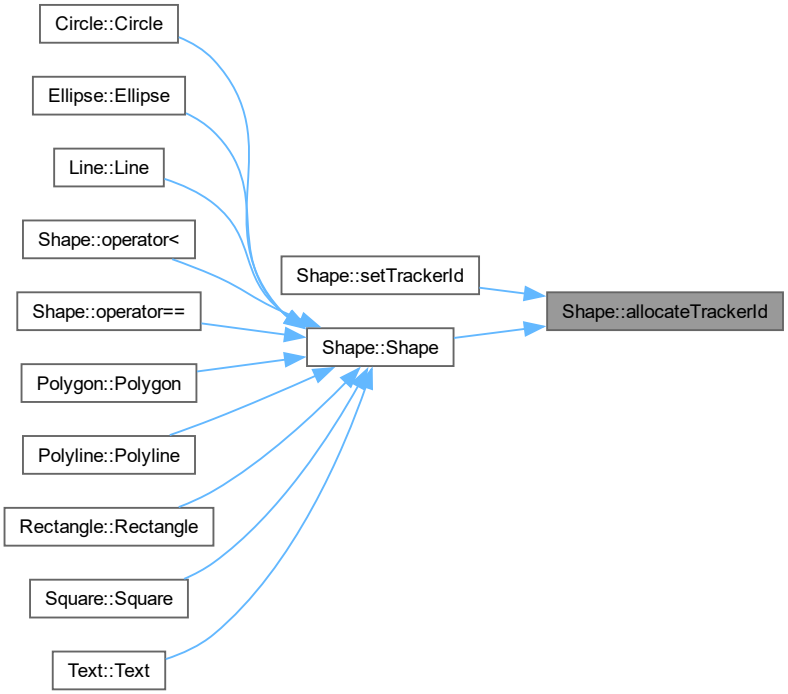
~Shape Destructor

8.17.3 Member Function Documentation

8.17.3.1 allocateTrackerId()

```
void Shape::allocateTrackerId (
            int shapeId)
```

Here is the caller graph for this function:



8.17.3.2 Area()

```
virtual double Shape::Area () const  [pure virtual]
```

Area - Returns the area of the shape.

Returns

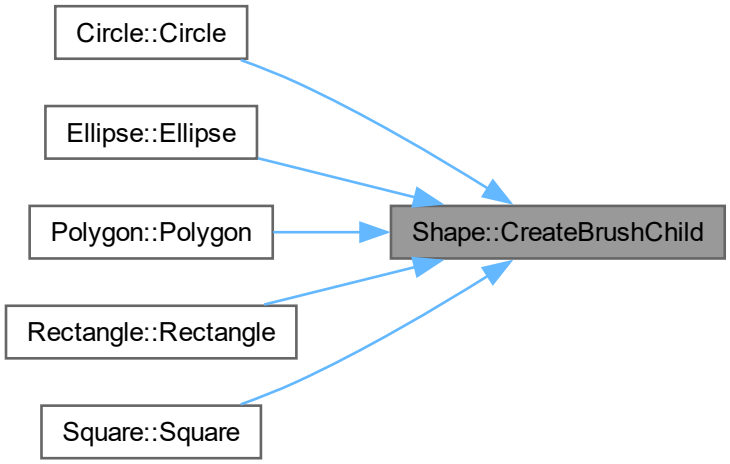
Implemented in Circle, Ellipse, Line, Polygon, Polyline, Rectangle, Square, and Text.

8.17.3.3 CreateBrushChild()

```
void Shape::CreateBrushChild ()
```

CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.

Here is the caller graph for this function:

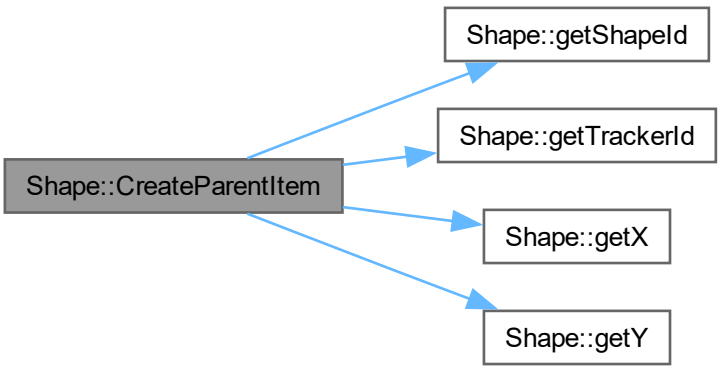


8.17.3.4 CreateParentItem()

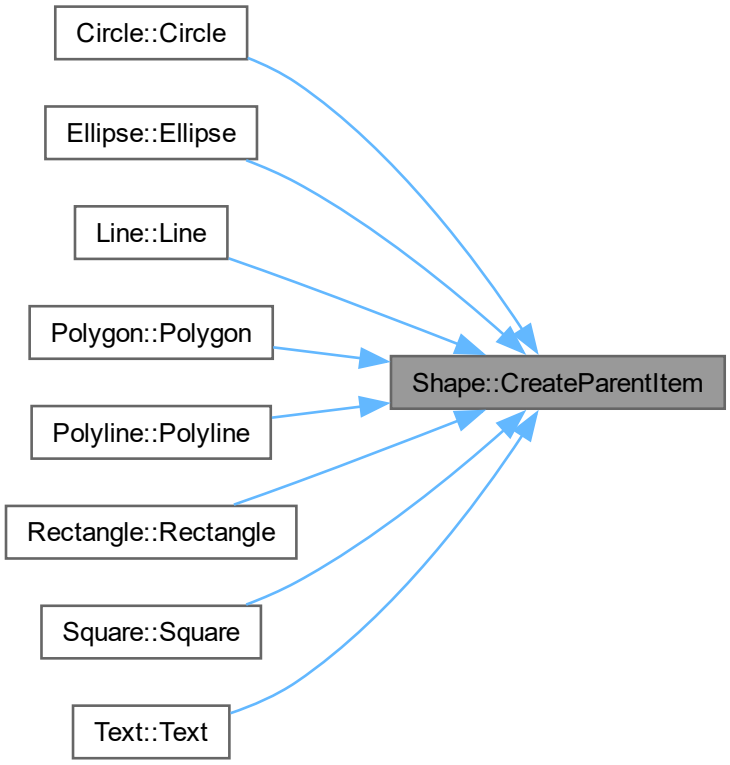
```
void Shape::CreateParentItem ()
```

CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.

Here is the call graph for this function:



Here is the caller graph for this function:

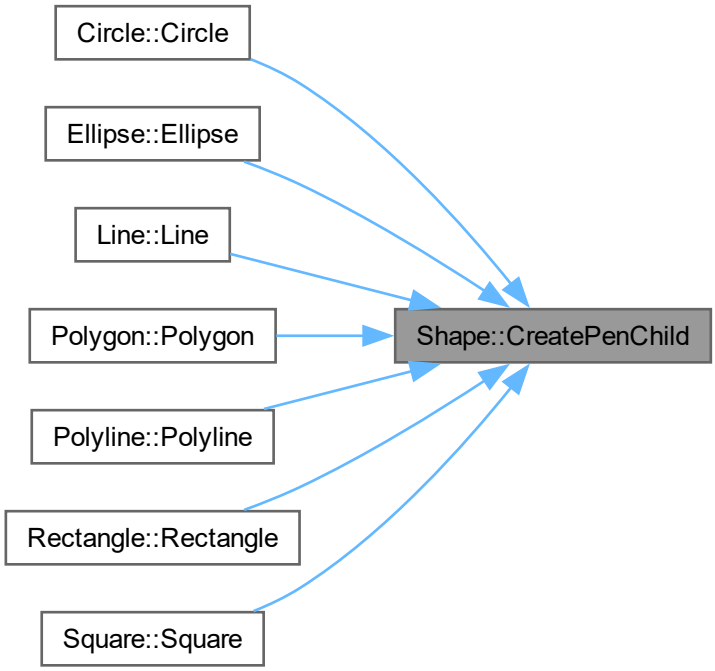


8.17.3.5 CreatePenChild()

```
void Shape::CreatePenChild ()
```

CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.

Here is the caller graph for this function:



8.17.3.6 CreatePointsChild()

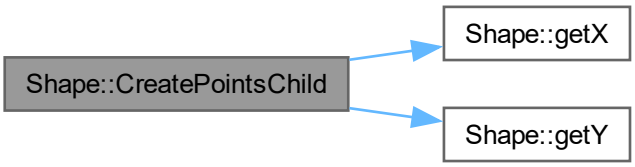
```
void Shape::CreatePointsChild (
            const int POINTS_NUM)
```

CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.

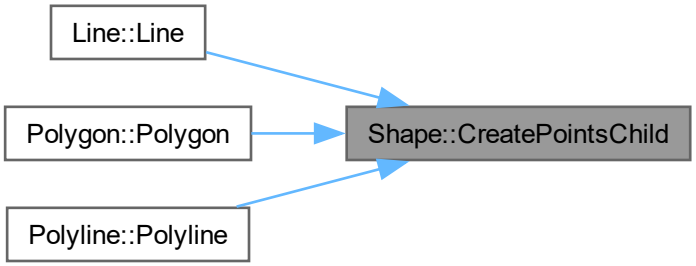
Parameters

POINTS_NUM	- Number of points being added

Here is the call graph for this function:



Here is the caller graph for this function:



8.17.3.7 Draw()

```
virtual void Shape::Draw (
            QWidget * renderArea)  [pure virtual]
```

Draw - Draws the shape to the associated renderArea.

Parameters

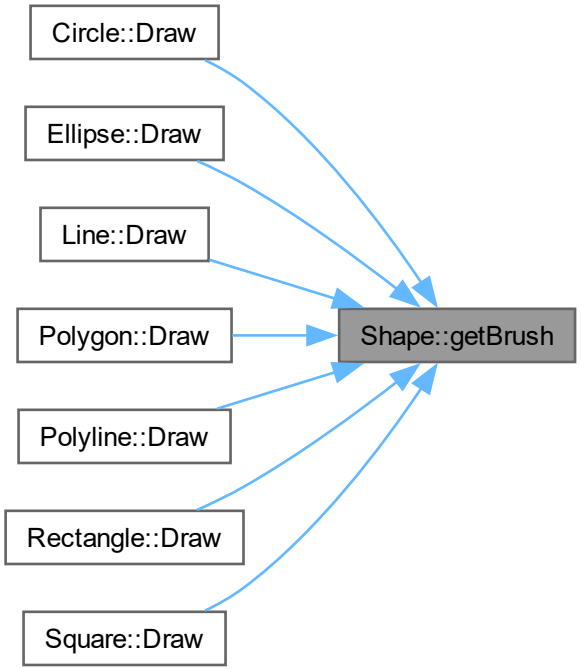
renderArea	- QWidget to be drawn on

Implemented in Circle, Ellipse, Line, Polygon, Polyline, Rectangle, Square, and Text.

8.17.3.8 getBrush()

```
QBrush Shape::getBrush () const
```

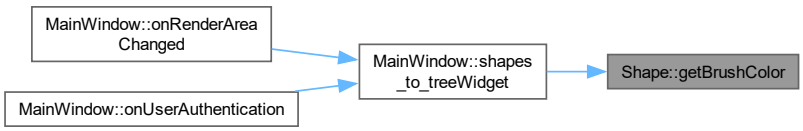
Here is the caller graph for this function:



8.17.3.9 getBrushColor()

```
QColor Shape::getBrushColor () const
```

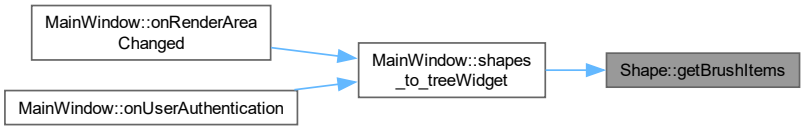
Here is the caller graph for this function:



8.17.3.10 getBrushItems()

```
alpha::vector< QTreeWidgetItem * > & Shape::getBrushItems ()
```

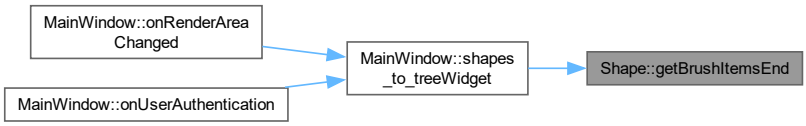
Here is the caller graph for this function:



8.17.3.11 getBrushItemsEnd()

```
int Shape::getBrushItemsEnd () const
```

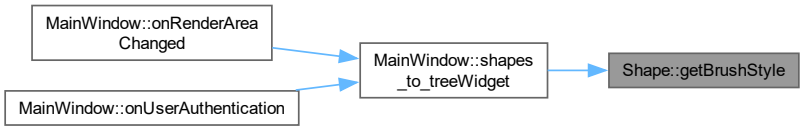
Returns the dereferenced penItems.end() - 1 for the last initialized element of the vector Here is the caller graph for this function:



8.17.3.12 getBrushStyle()

```
BrushStyle Shape::getBrushStyle () const
```

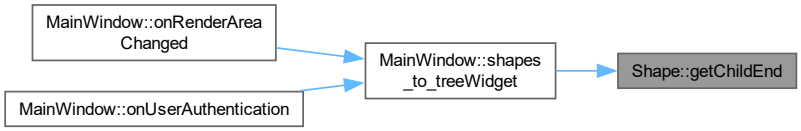
Here is the caller graph for this function:



8.17.3.13 getChildEnd()

```
int Shape::getChildEnd () const
```

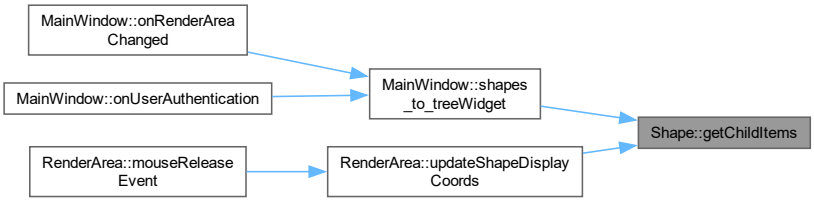
Here is the caller graph for this function:



8.17.3.14 getChildItems()

```
alpha::vector< QTreeWidgetItem * > & Shape::getChildItems ()
```

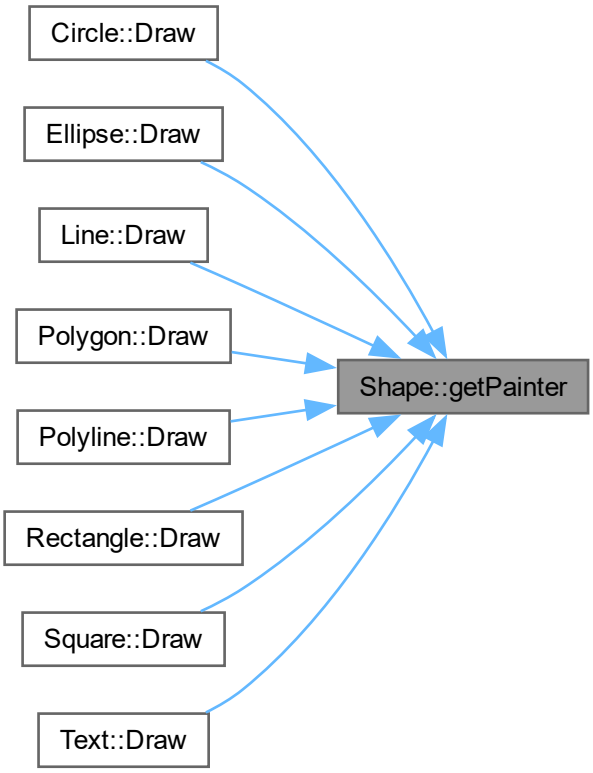
Here is the caller graph for this function:



8.17.3.15 getPainter()

```
QPainter & Shape::getPainter ()
```

Here is the caller graph for this function:



8.17.3.16 getParentItem()

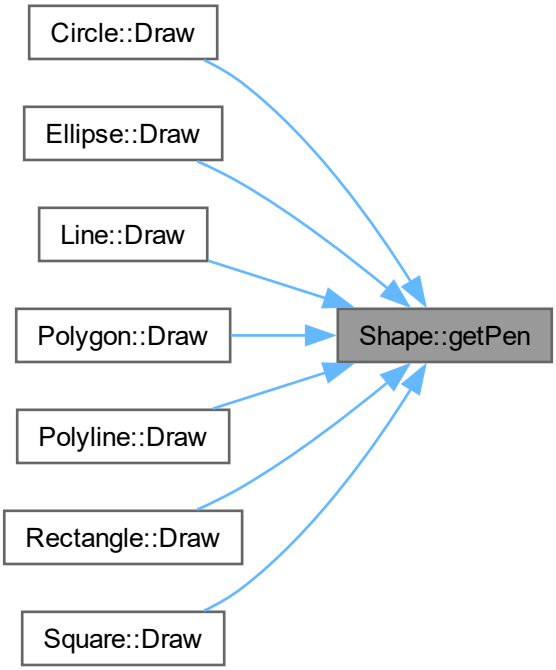
```
QTreeWidgetItem * Shape::getParentItem ()
```

Here is the caller graph for this function:

8.17.3.17 getPen()

```
QPen Shape::getPen () const
```

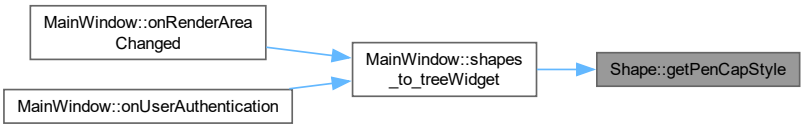
Here is the caller graph for this function:



8.17.3.18 getPenCapStyle()

```
PenCapStyle Shape::getPenCapStyle () const
```

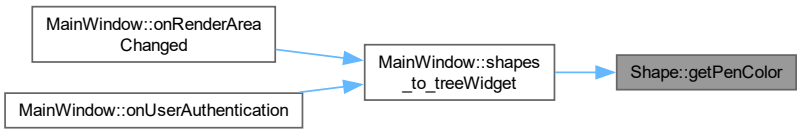
Here is the caller graph for this function:



8.17.3.19 getPenColor()

```
QColor Shape::getPenColor () const
```

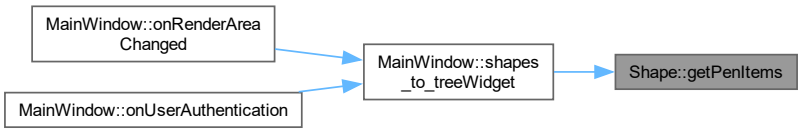
Here is the caller graph for this function:



8.17.3.20 getPenItems()

```
alpha::vector< QTreeWidgetItem * > & Shape::getPenItems ()
```

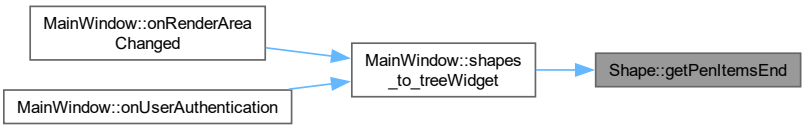
Here is the caller graph for this function:



8.17.3.21 getPenItemsEnd()

```
int Shape::getPenItemsEnd () const
```

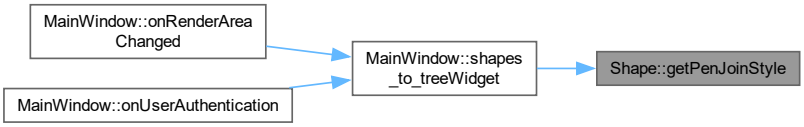
Returns the dereferenced childItems.end() - 1 for the last initialized element of the vector Here is the caller graph for this function:



8.17.3.22 getPenJoinStyle()

```
PenJoinStyle Shape::getPenJoinStyle () const
```

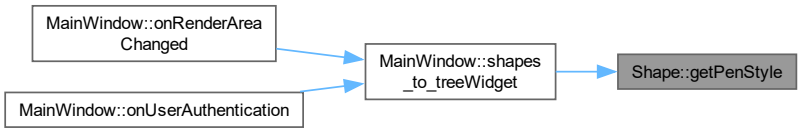
Here is the caller graph for this function:



8.17.3.23 getPenStyle()

```
PenStyle Shape::getPenStyle () const
```

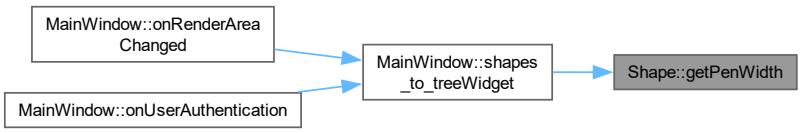
Here is the caller graph for this function:



8.17.3.24 getPenWidth()

```
int Shape::getPenWidth () const
```

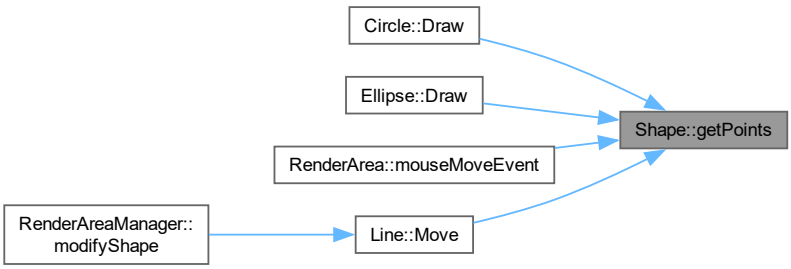
Here is the caller graph for this function:



8.17.3.25 getPoints()

```
QPoint Shape::getPoints () const
```

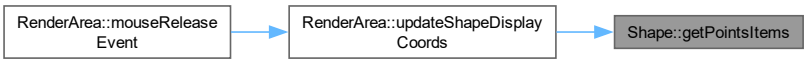
Here is the caller graph for this function:



8.17.3.26 getPointsItems()

```
alpha::vector< QTreeWidgetItem * > & Shape::getPointsItems ()
```

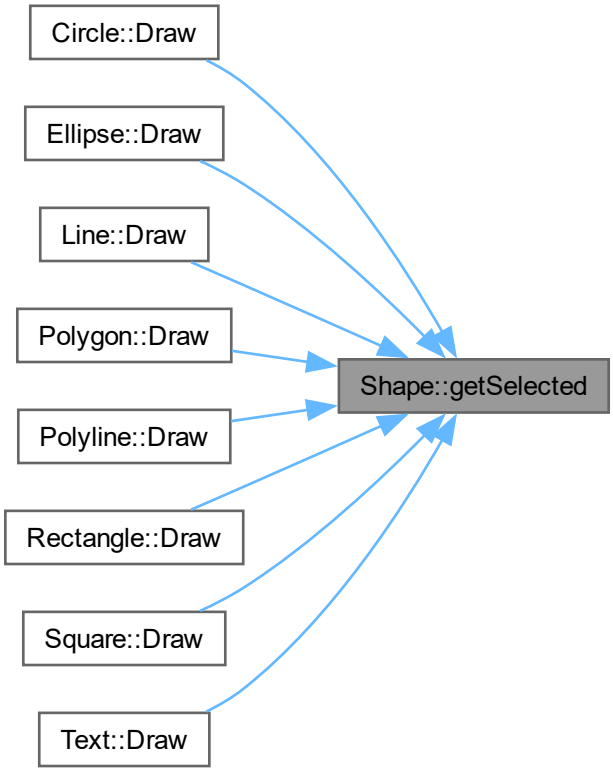
Here is the caller graph for this function:



8.17.3.27 getSelected()

```
bool Shape::getSelected () const
```

Here is the caller graph for this function:

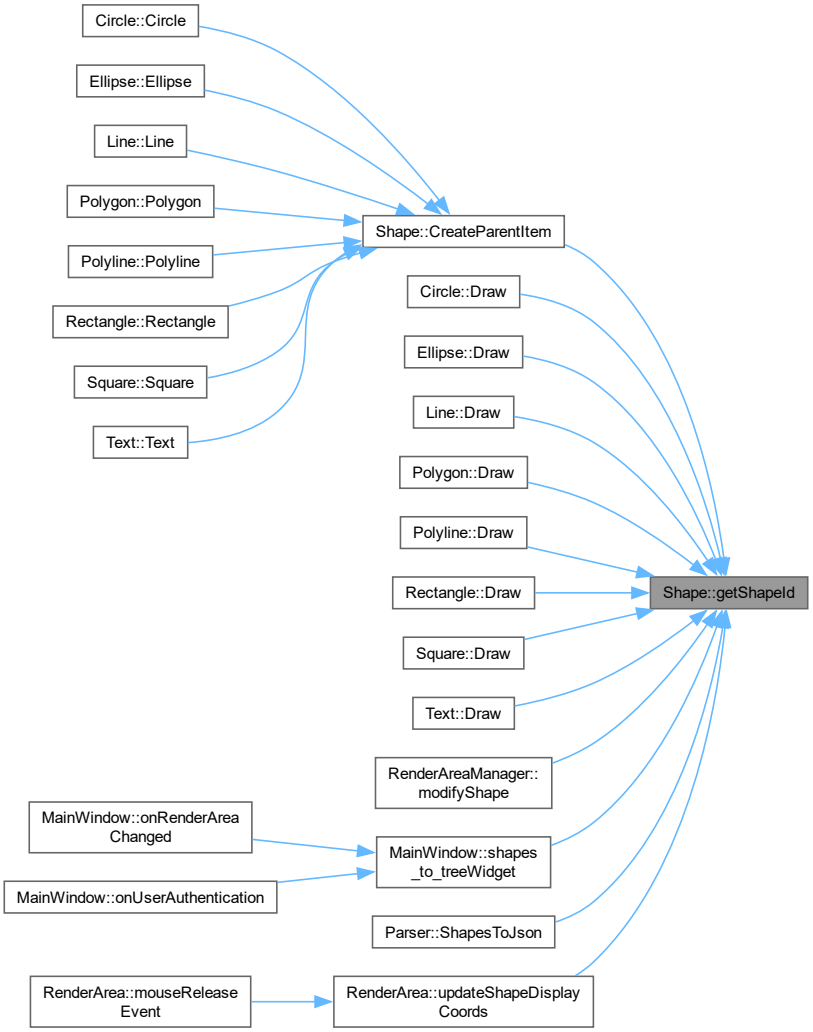


8.17.3.28 getShapeId()

```
int Shape::getShapeId () const
```

Accessor Functions - Returns the data named after them.

Here is the caller graph for this function:



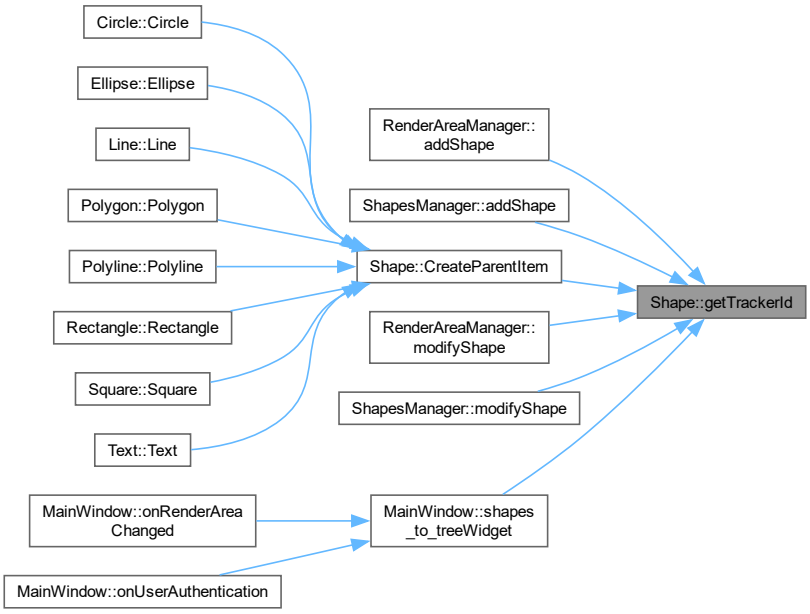
8.17.3.29 getShapeType()

```
string Shape::getShapeType () const
```

8.17.3.30 getTrackerId()

```
int Shape::getTrackerId () const
```

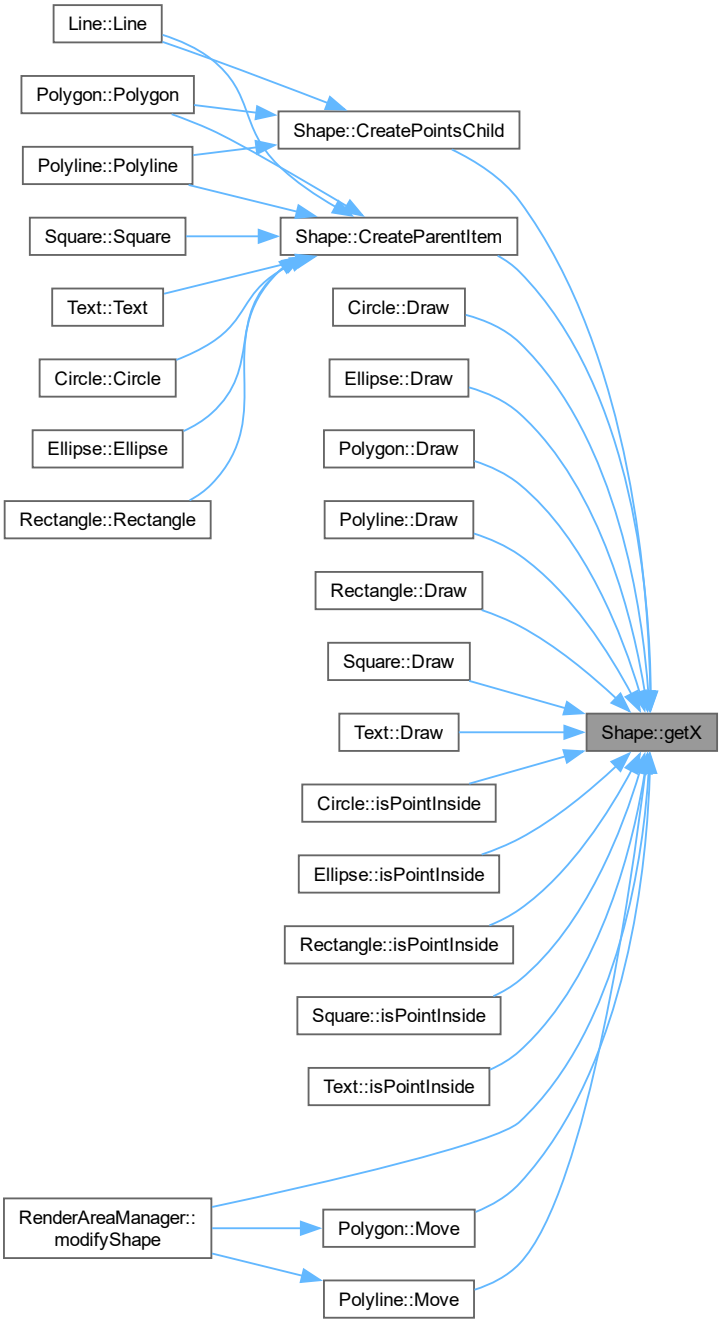
Here is the caller graph for this function:



8.17.3.31 getX()

```
int Shape::getX () const
```

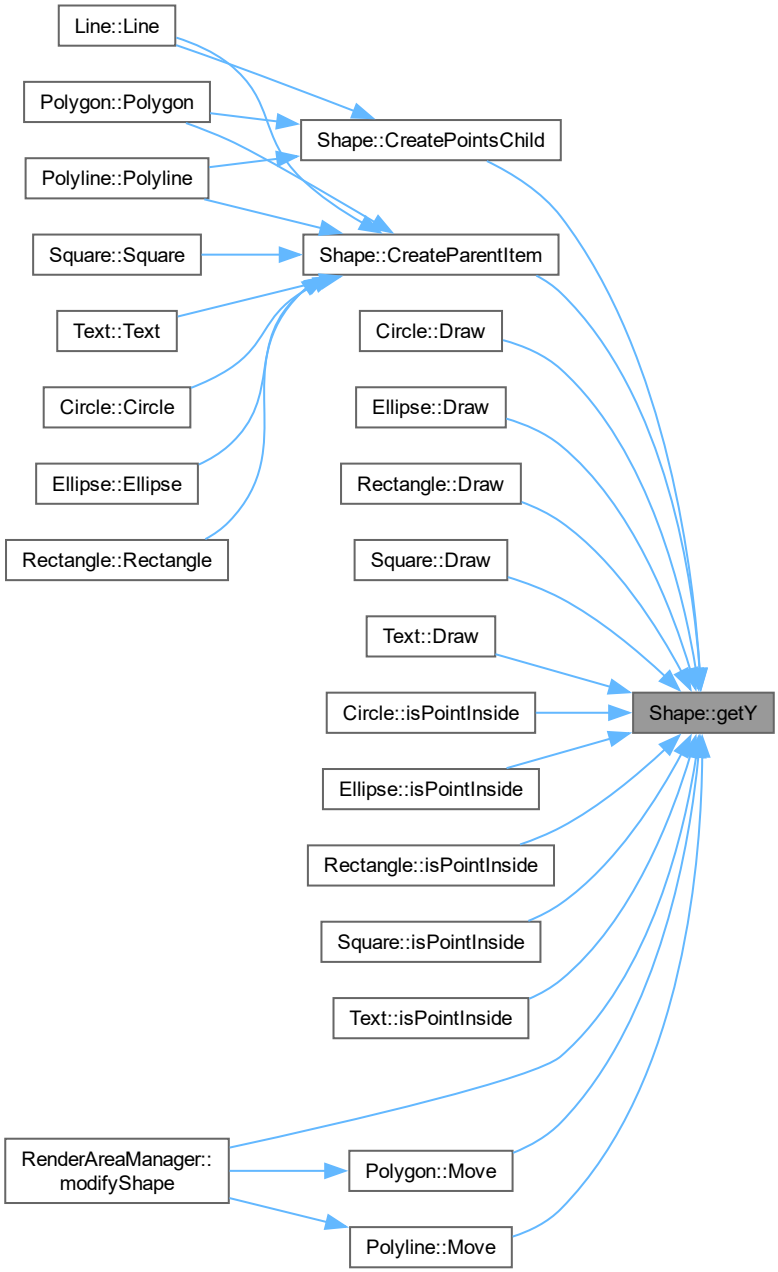
Here is the caller graph for this function:



8.17.3.32 getY()

```
int Shape::getY () const
```

Here is the caller graph for this function:



8.17.3.33 isPointInside()

```
virtual bool Shape::isPointInside (
            const QPoint & point) const  [pure virtual]
```

isPointInside - Returns true if point is inside the shape

Parameters

point	- QPoint being checked

Returns

Implemented in Circle, Ellipse, Line, Polygon, Polyline, Rectangle, Square, and Text.

8.17.3.34 Move()

```
void Shape::Move (
            int x,
            int y)  [virtual]
```

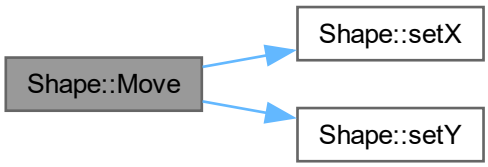
Move - Moves the shape to the x and y coords.

Parameters

x	- x coordinate
y	- y coordinate

Implemented in Line, Polygon, and Polyline.

Here is the call graph for this function:



Here is the caller graph for this function:

8.17.3.35 Perimeter()

```
virtual double Shape::Perimeter () const  [pure virtual]
```

Perimeter - Returns the perimeter of the shape.

Returns

Implemented in Circle, Ellipse, Line, Polygon, Polyline, Rectangle, Square, and Text.

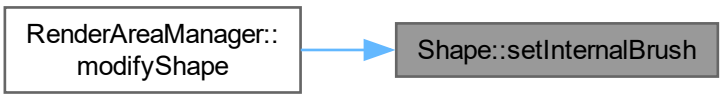
8.17.3.36 setBrush()

```
void Shape::setBrush (
            GlobalColor brushColor,
            BrushStyle brushStyle)
```

8.17.3.37 setInternalBrush()

```
QBrush & Shape::setInternalBrush ()
```

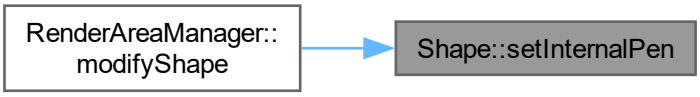
Here is the caller graph for this function:



8.17.3.38 setInternalPen()

```
QPen & Shape::setInternalPen ()
```

Here is the caller graph for this function:



8.17.3.39 setPen()

```
void Shape::setPen (
            GlobalColor penColor,
            int penWidth,
            PenStyle penStyle,
            PenCapStyle penCapStyle,
            PenJoinStyle penJoinStyle)
```

8.17.3.40 setSelected()

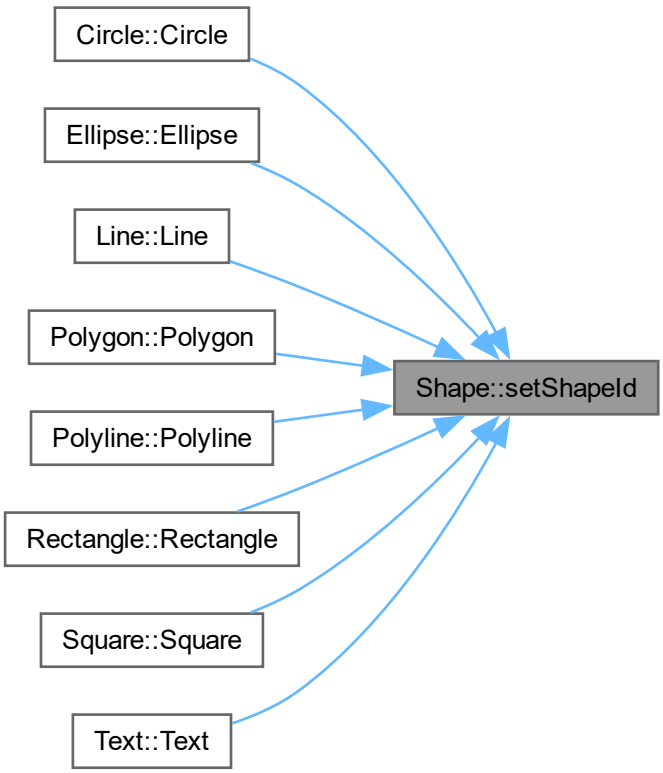
```
void Shape::setSelected (
            bool selected)
```

8.17.3.41 setShapeId()

```
void Shape::setShapeId (
            int shapeId)
```

Accessor Functions.

Returns the dereferenced brushItems.end() - 1 for the last initialized element of the vector Mutator Functions - Sets the data of the item to the passed param Here is the caller graph for this function:



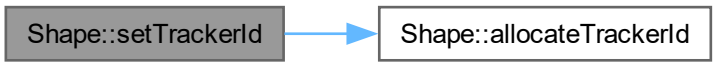
8.17.3.42 setShapeType()

```
void Shape::setShapeType (
            string shapeType)
```

8.17.3.43 setTrackerId()

```
void Shape::setTrackerId (
            int trackerId)
```

Here is the call graph for this function:

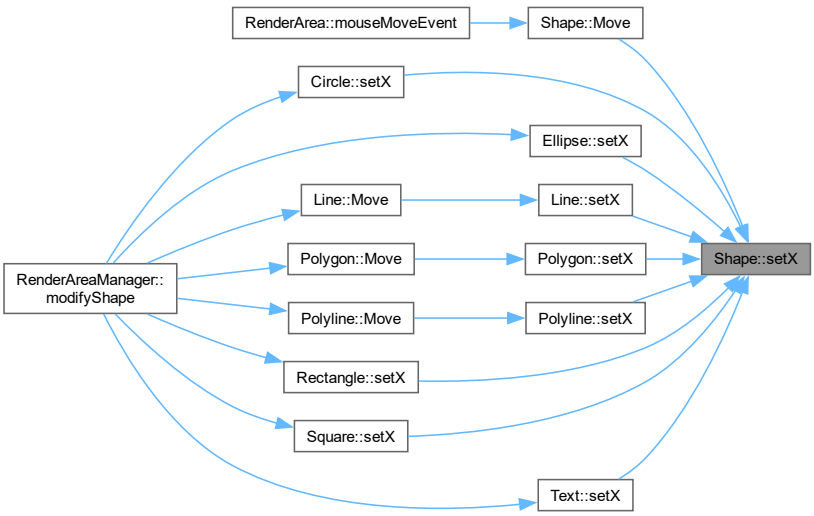


8.17.3.44 setX()

```
void Shape::setX (
            int x)
```

Implemented in Circle, Ellipse, Line, Polygon, Polyline, Rectangle, Square, and Text.

Here is the caller graph for this function:

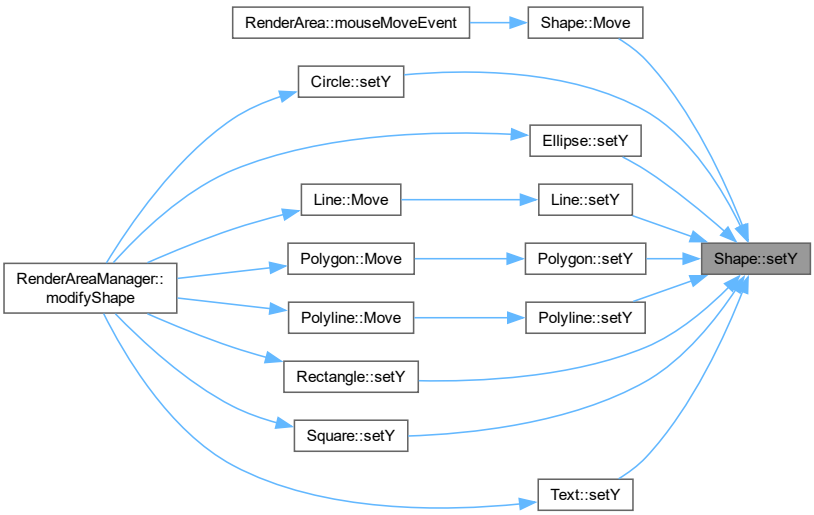


8.17.3.45 setY()

```
void Shape::setY (
            int y)
```

Implemented in Circle, Ellipse, Line, Polygon, Polyline, Rectangle, Square, and Text.

Here is the caller graph for this function:



8.17.4 Friends And Related Symbol Documentation

8.17.4.1 operator<

```
bool operator< (
            const Shape & shape1,
            const Shape & shape2)  [friend]
```

operator < - Overloaded less than operator for comparing two shapeId's

Parameters

shape1	
shape2	

Returns

8.17.4.2 operator==

```
bool operator== (
            const Shape & shape1,
            const Shape & shape2)  [friend]
```

operator == - Overloaded equality operator for comparing two shapeId's

Parameters

shape1	
shape2	

Returns

8.17.5 Member Data Documentation

8.17.5.1 brushItems

```
alpha::vector<QTreeWidgetItem*> Shape::brushItems  [protected]
```

vector of QTreeWidgetItem* holding data of all brush items

8.17.5.2 childItems

```
alpha::vector<QTreeWidgetItem*> Shape::childItems  [protected]
```

vector of QTreeWidgetItem* holding data of all child items in parentItem

8.17.5.3 nextTracker

```
int Shape::nextTracker = {}  [static]
```

8.17.5.4 parentItem

```
QTreeWidgetItem* Shape::parentItem  [protected]
```

Mutator Functions.

QTreeWidgetItem* holding treeWidget data of each shape

8.17.5.5 penItems

```
alpha::vector<QTreeWidgetItem*> Shape::penItems  [protected]
```

vector of QTreeWidgetItem* holding data of all pen items

8.17.5.6 pointsItems

```
alpha::vector<QTreeWidgetItem*> Shape::pointsItems  [protected]
```

vector of QTreeWidgetItem* holding data of all points (besides coords) in parentItem

8.17.5.7 trackersInUse

```
bool Shape::trackersInUse = {}  [static]
```

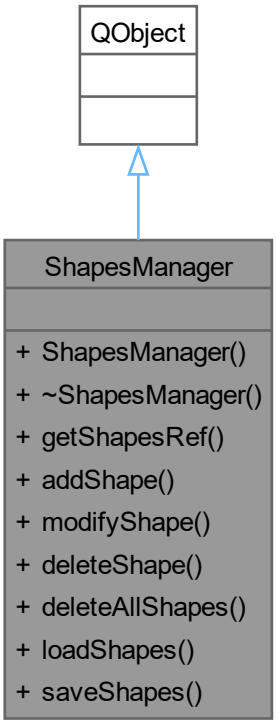
The documentation for this interface was generated from the following files:

- src/code/shape.h
- src/code/shape.cpp

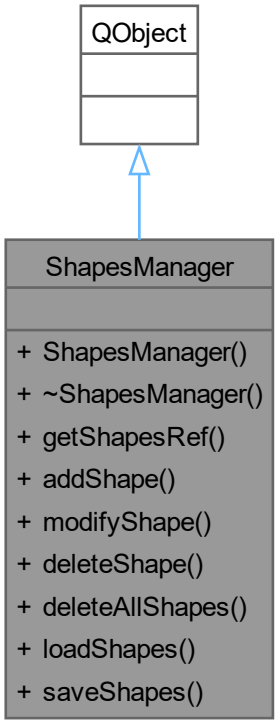
8.18 ShapesManager Class Reference

```
#include <ShapesManager.h>
```

Inheritance diagram for ShapesManager:



Collaboration diagram for ShapesManager:



Signals

- void shapesChanged ()
- void shapesNotChanged (const QString &messsage)
- void statusMessage (const QString &message)

Public Member Functions

- ShapesManager (QObject *parent=nullptr)

  Default Constructor.
- ~ShapesManager ()
- alpha::vector< Shape * > * getShapesRef ()

  Returns the shapes vector as a reference.
- void addShape (Shape *shape)

  These functions are used to add, delete, and load shapes in the ShapesManager.
- void modifyShape (Shape *shape)
- void deleteShape (const int trackerId)
- void deleteAllShapes ()
- void loadShapes ()
- void saveShapes ()

8.18.1 Constructor & Destructor Documentation

8.18.1.1 ShapesManager()

```
ShapesManager::ShapesManager (
            QObject * parent = nullptr)  [explicit]
```

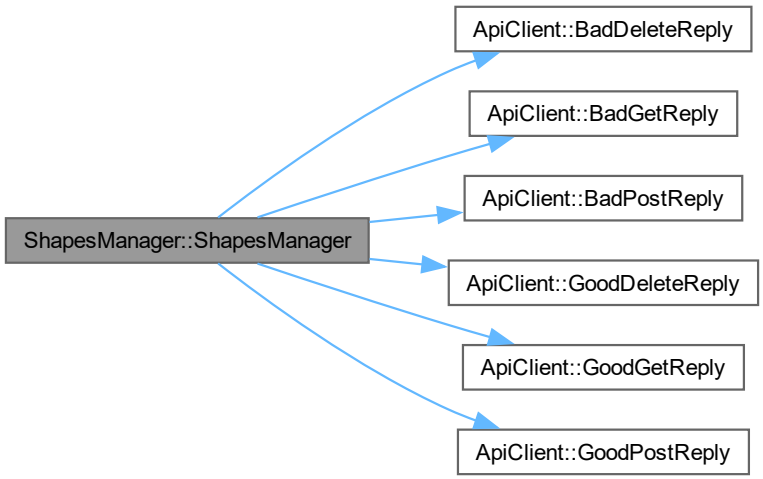
Default Constructor.

This constructor initializes the ShapesManager object and connects to the ApiClient signals for handling shape data.

Parameters

parent	- any QObject to tie this instantiation to ensure automatic deletion

Here is the call graph for this function:



8.18.1.2 ~ShapesManager()

```
ShapesManager::~ShapesManager ()
```

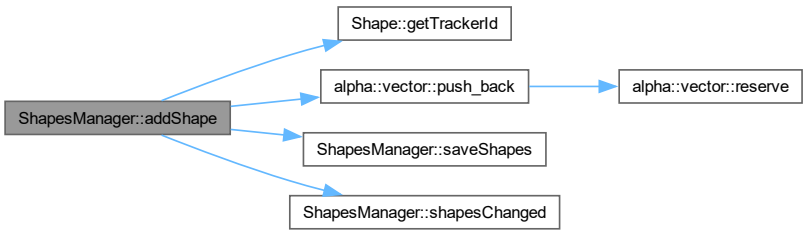
8.18.2 Member Function Documentation

8.18.2.1 addShape()

```
void ShapesManager::addShape (
            Shape * shape)
```

These functions are used to add, delete, and load shapes in the ShapesManager.

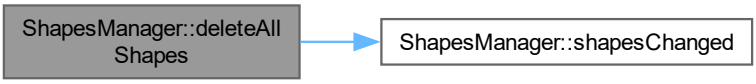
All of these functions emit the shapesChanged() signal to notify the frontend that the shapes have changed and need to be redrawn except for saveShapes(). Here is the call graph for this function:



8.18.2.2 deleteAllShapes()

```
void ShapesManager::deleteAllShapes ()
```

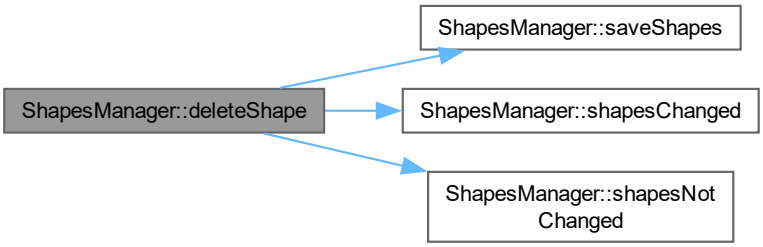
Here is the call graph for this function:



8.18.2.3 deleteShape()

```
void ShapesManager::deleteShape (
            const int trackerId)
```

Here is the call graph for this function:



8.18.2.4 getShapesRef()

```
alpha::vector< Shape * > * ShapesManager::getShapesRef ()
```

Returns the shapes vector as a reference.

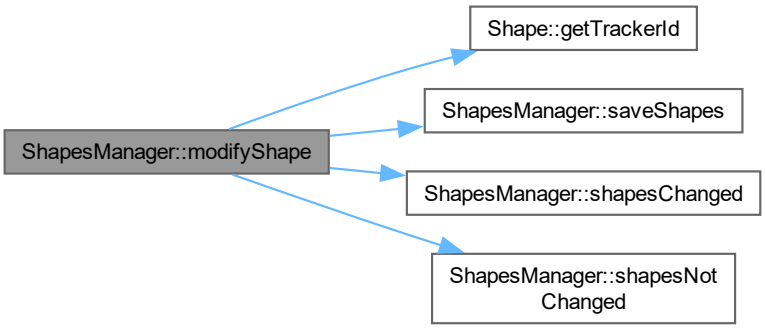
8.18.2.5 loadShapes()

```
void ShapesManager::loadShapes ()
```

8.18.2.6 modifyShape()

```
void ShapesManager::modifyShape (
            Shape * shape)
```

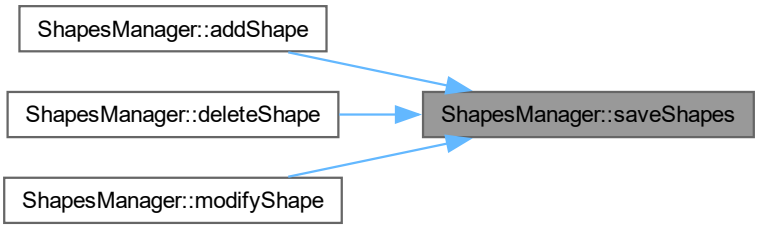
Here is the call graph for this function:



8.18.2.7 saveShapes()

```
void ShapesManager::saveShapes ()
```

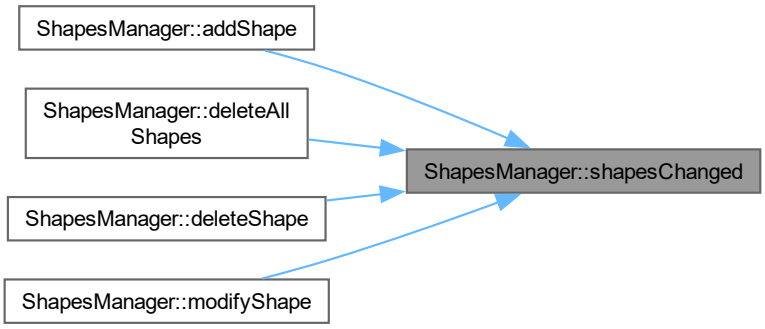
Here is the caller graph for this function:



8.18.2.8 shapesChanged

```
void ShapesManager::shapesChanged ()  [signal]
```

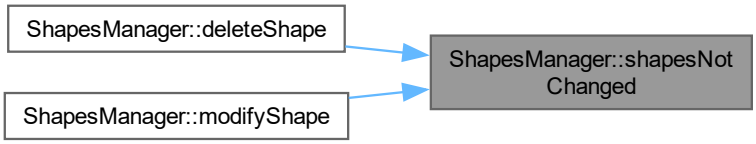
These signals are meant to connect to the Shapes window in the frontend. -shapesChanged(): signal for when the shapes vector is changed in any way so that the frontend knows to refresh the window and redraw the shapes -statusMessage(): signal for slot functions below. It passes the message to the frontend so it can display a popup saying the shapes were saved successfully, or whatever the message is for the user. Here is the caller graph for this function:



8.18.2.9 shapesNotChanged

```
void ShapesManager::shapesNotChanged (
            const QString & messsage)  [signal]
```

Here is the caller graph for this function:



8.18.2.10 statusMessage

```
void ShapesManager::statusMessage (
            const QString & message)  [signal]
```

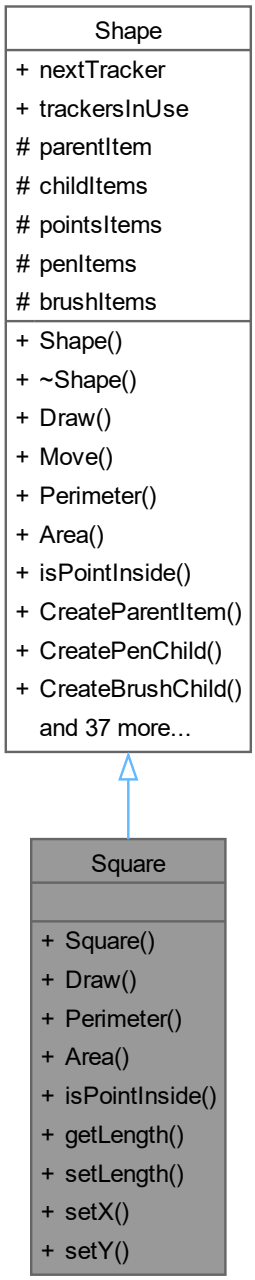
The documentation for this class was generated from the following files:

- src/code/ShapesManager.h
- src/code/ShapesManager.cpp

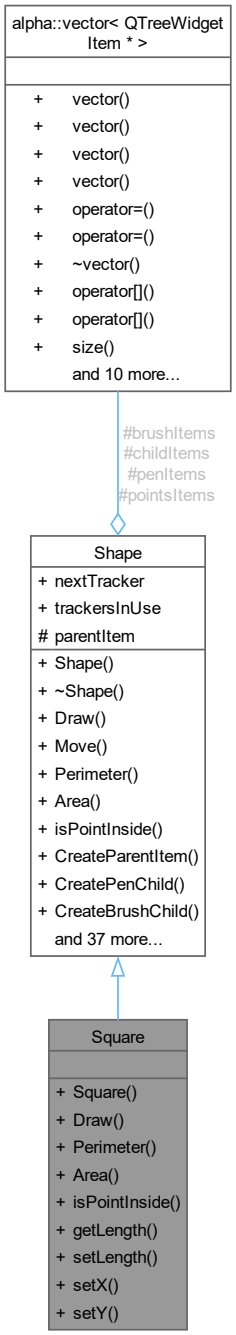
8.19 Square Class Reference

`#include <square.h>`

Inheritance diagram for Square:



Collaboration diagram for Square:



Public Member Functions

- Square (string shapeType, QPoint coords, QPen pen, QBrush brush, int length)
- void Draw (QWidget ∗renderArea) override

  Draw - Draws the shape to the associated renderArea.
- double Perimeter () const override

  Perimeter - Returns the perimeter of the shape.

- double Area () const override

  Area - Returns the area of the shape.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns true if point is inside the shape
- int getLength () const
- void setLength (int newLength)
- void setX (int newX)
- void setY (int newY)

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- virtual void Move (int x, int y)

  Move - Moves the shape to the x and y coords.
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const
- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

Accessor Functions.

- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem ∗ parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem ∗ > childItems

  vector of QTreeWidgetItem∗ holding data of all child items in parentItem
- alpha::vector< QTreeWidgetItem ∗ > pointsItems

  vector of QTreeWidgetItem∗ holding data of all points (besides coords) in parentItem
- alpha::vector< QTreeWidgetItem ∗ > penItems

  vector of QTreeWidgetItem∗ holding data of all pen items
- alpha::vector< QTreeWidgetItem ∗ > brushItems

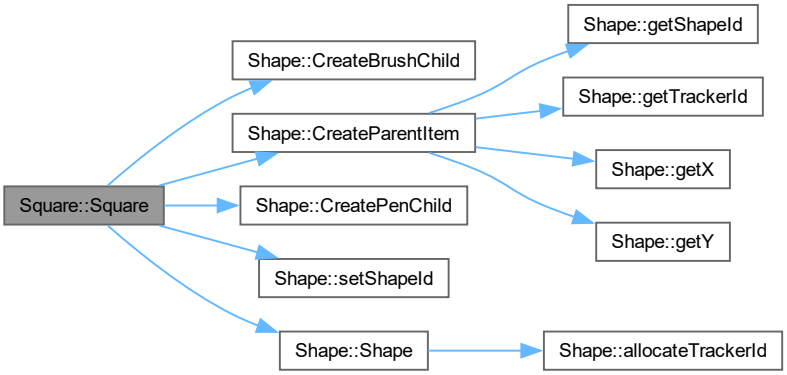
  vector of QTreeWidgetItem∗ holding data of all brush items

8.19.1 Constructor & Destructor Documentation

8.19.1.1 Square()

```
Square::Square (
            string shapeType,
            QPoint coords,
            QPen pen,
            QBrush brush,
            int length)
```

Here is the call graph for this function:



8.19.2 Member Function Documentation

8.19.2.1 Area()

```
double Square::Area () const  [override], [virtual]
```

Area - Returns the area of the shape.

Returns

Implements Shape.

8.19.2.2 Draw()

```
void Square::Draw (
            QWidget * renderArea)  [override], [virtual]
```

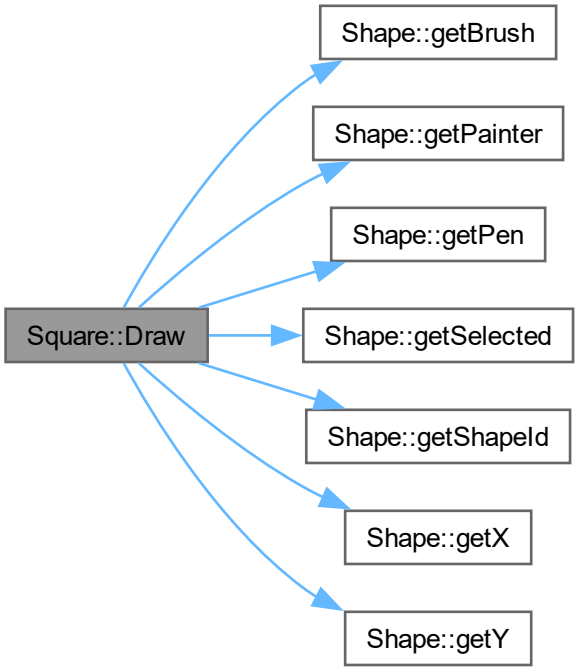
Draw - Draws the shape to the associated renderArea.

Parameters

renderArea	- QWidget to be drawn on

Implements Shape.

Here is the call graph for this function:



8.19.2.3 getLength()

```
int Square::getLength () const
```

8.19.2.4 isPointInside()

```
bool Square::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns true if point is inside the shape

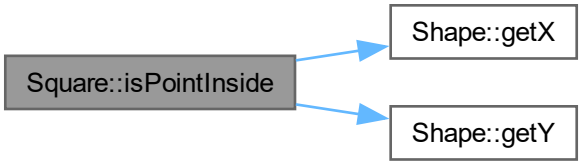
Parameters

point	- QPoint being checked

Returns

Implements Shape.

Here is the call graph for this function:



8.19.2.5 Perimeter()

```
double Square::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the shape.

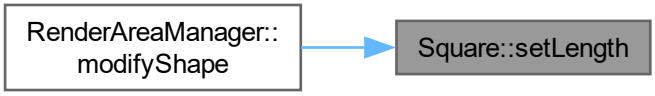
Returns

Implements Shape.

8.19.2.6 setLength()

```
void Square::setLength (
            int newLength)
```

Here is the caller graph for this function:

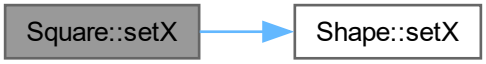


8.19.2.7 setX()

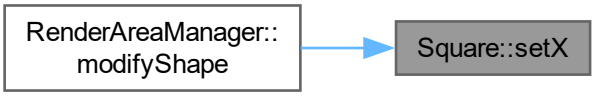
```
void Square::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:

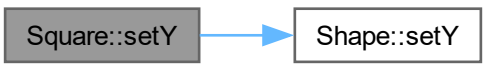


8.19.2.8 setY()

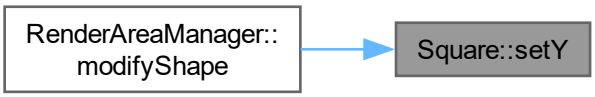
```
void Square::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



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

- src/code/square.h
- src/code/square.cpp

8.20 Testimonial Class Reference

`#include <Testimonial.h>`

Collaboration diagram for Testimonial:

Public Member Functions

- Testimonial (const QString &author="", const QString &content="", bool isGuest=true)
- QString getAuthor () const
- QString getContent () const
- QDateTime getTimestamp () const
- bool isGuest () const
- bool isSatisfactory () const
- void setIsSatisfactory (bool value)
- QJsonObject toJson () const

Static Public Member Functions

- static Testimonial fromJson (const QJsonObject &json)

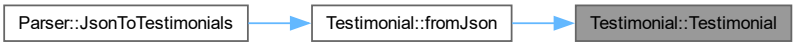
8.20.1 Constructor & Destructor Documentation

8.20.1.1 Testimonial()

```
Testimonial::Testimonial (
            const QString & author = "",
            const QString & content = "",
            bool isGuest = true)
```

Here is the call graph for this function:

Here is the caller graph for this function:

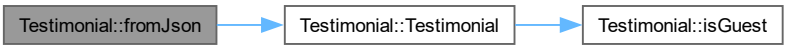


8.20.2 Member Function Documentation

8.20.2.1 fromJson()

```
Testimonial Testimonial::fromJson (
            const QJsonObject & json)  [static]
```

Here is the call graph for this function:



Here is the caller graph for this function:

8.20.2.2 getAuthor()

```
QString Testimonial::getAuthor () const  [inline]
```

8.20.2.3 getContent()

```
QString Testimonial::getContent () const  [inline]
```

8.20.2.4 getTimestamp()

```
QDateTime Testimonial::getTimestamp () const  [inline]
```

8.20.2.5 isGuest()

```
bool Testimonial::isGuest () const  [inline]
```

Here is the caller graph for this function:

8.20.2.6 isSatisfactory()

```
bool Testimonial::isSatisfactory () const  [inline]
```

8.20.2.7 setIsSatisfactory()

```
void Testimonial::setIsSatisfactory (
            bool value)  [inline]
```

8.20.2.8 toJson()

```
QJsonObject Testimonial::toJson () const
```

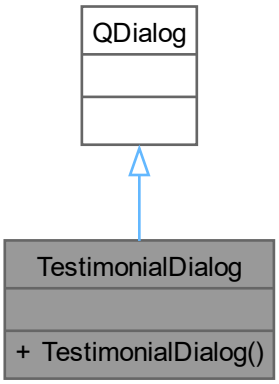
The documentation for this class was generated from the following files:

- src/code/Testimonial.h
- src/code/Testimonial.cpp

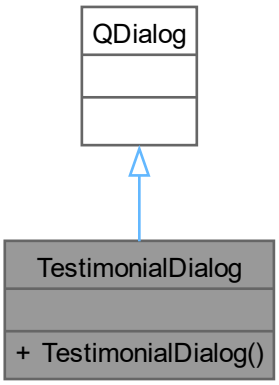
8.21 TestimonialDialog Class Reference

```
#include <TestimonialDialog.h>
```

Inheritance diagram for TestimonialDialog:



Collaboration diagram for TestimonialDialog:



Public Member Functions

- TestimonialDialog (QWidget *parent=nullptr)

8.21.1 Constructor & Destructor Documentation

8.21.1.1 TestimonialDialog()

```
TestimonialDialog::TestimonialDialog (
            QWidget * parent = nullptr)  [explicit]
```

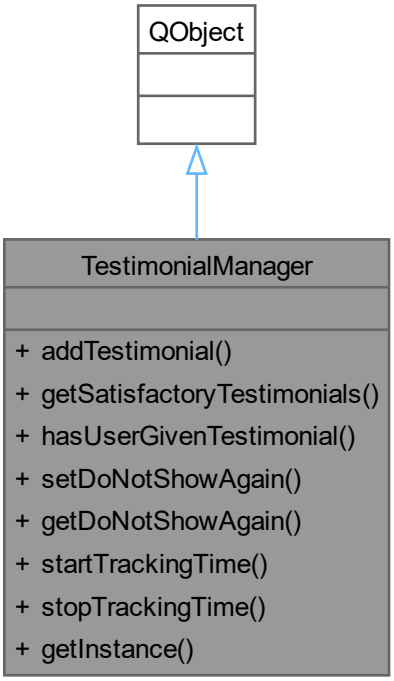
The documentation for this class was generated from the following files:

- src/code/TestimonialDialog.h
- src/code/TestimonialDialog.cpp

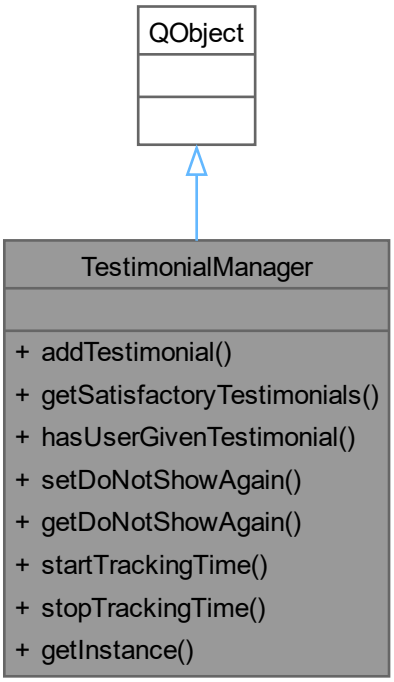
8.22 TestimonialManager Class Reference

```
#include <TestimonialManager.h>
```

Inheritance diagram for TestimonialManager:



Collaboration diagram for TestimonialManager:



Signals

- void shouldPromptForTestimonial ()

Public Member Functions

- void addTestimonial (const Testimonial &testimonial)
- QVector< Testimonial > getSatisfactoryTestimonials () const
- bool hasUserGivenTestimonial (const QString &username) const
- void setDoNotShowAgain (const QString &username, bool value)
- bool getDoNotShowAgain (const QString &username) const
- void startTrackingTime ()
- void stopTrackingTime ()

Static Public Member Functions

- static TestimonialManager & getInstance ()

8.22.1 Member Function Documentation

8.22.1.1 addTestimonial()

```
void TestimonialManager::addTestimonial (
            const Testimonial & testimonial)
```

8.22.1.2 getDoNotShowAgain()

```
bool TestimonialManager::getDoNotShowAgain (
            const QString & username) const
```

8.22.1.3 getInstance()

```
TestimonialManager & TestimonialManager::getInstance ()  [static]
```

8.22.1.4 getSatisfactoryTestimonials()

```
QVector< Testimonial > TestimonialManager::getSatisfactoryTestimonials () const
```

8.22.1.5 hasUserGivenTestimonial()

```
bool TestimonialManager::hasUserGivenTestimonial (
            const QString & username) const
```

8.22.1.6 setDoNotShowAgain()

```
void TestimonialManager::setDoNotShowAgain (
            const QString & username,
            bool value)
```

8.22.1.7 shouldPromptForTestimonial

```
void TestimonialManager::shouldPromptForTestimonial ()  [signal]
```

8.22.1.8 startTrackingTime()

```
void TestimonialManager::startTrackingTime ()
```

8.22.1.9 stopTrackingTime()

```
void TestimonialManager::stopTrackingTime ()
```

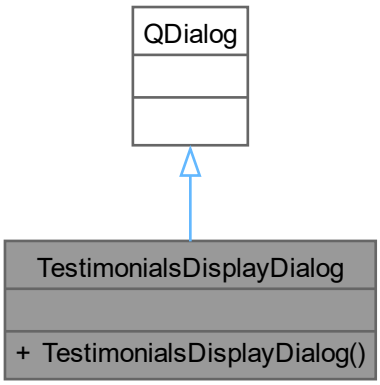
The documentation for this class was generated from the following files:

- src/code/TestimonialManager.h
- src/code/TestimonialManager.cpp

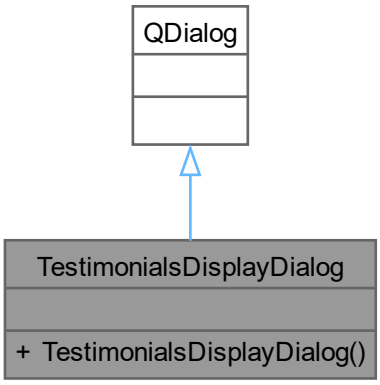
8.23 TestimonialsDisplayDialog Class Reference

`#include <TestimonialsDisplayDialog.h>`

Inheritance diagram for TestimonialsDisplayDialog:



Collaboration diagram for TestimonialsDisplayDialog:



Public Member Functions

- TestimonialsDisplayDialog (QWidget *parent=nullptr)

8.23.1 Constructor & Destructor Documentation

8.23.1.1 TestimonialsDisplayDialog()

```
TestimonialsDisplayDialog::TestimonialsDisplayDialog (
            QWidget * parent = nullptr)  [explicit]
```

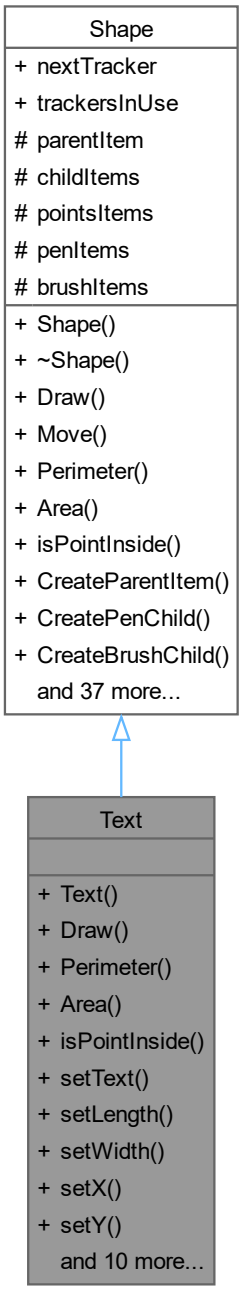
The documentation for this class was generated from the following files:

- src/code/TestimonialsDisplayDialog.h
- src/code/TestimonialsDisplayDialog.cpp

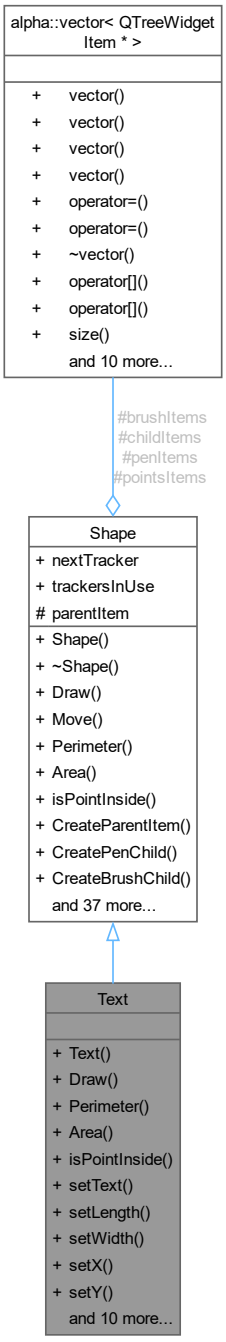
8.24 Text Class Reference

```
#include <text.h>
```

Inheritance diagram for Text:



Collaboration diagram for Text:



Public Member Functions

- Text (string shapeType, QPoint coords, QString textString, GlobalColor textColor, AlignmentFlag text↩ Alignment, QFont font, int length, int width)
- void Draw (QWidget ∗renderArea) override

  Draw - Draws the shape to the associated renderArea.
- double Perimeter () const override

Perimeter - Returns the perimeter of the shape.

- double Area () const override

  Area - Returns the area of the shape.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns true if point is inside the shape
- void setText (QString text)
- void setLength (int newLength)
- void setWidth (int newWidth)
- void setX (int newX)
- void setY (int newY)
- void setAlignment (Qt::AlignmentFlag alignment)
- QFont & setInternalFont ()
- int getLength () const
- int getWidth () const
- QString getTextString () const
- GlobalColor getTextColor () const
- QFont getFont () const
- AlignmentFlag getTextAlignment () const
- int getFontStyle () const
- QFont::Weight getFontWeight () const

Public Member Functions inherited from Shape

- Shape (string shapeType, QPoint coords, QPen pen, QBrush brush)

  Shape Constructor.
- virtual ~Shape ()

  ~Shape Destructor
- virtual void Move (int x, int y)

  Move - Moves the shape to the x and y coords.
- void CreateParentItem ()

  CreateParentItem - Adds data cooresponding to all shapes to parentItem for a QTreeWidget.
- void CreatePenChild ()

  CreatePenChild - Adds pen data to penItems vector for a QTreeWidget.
- void CreateBrushChild ()

  CreateBrushChild - Adds brush data to brushItems vector for a QTreeWidget.
- void CreatePointsChild (const int POINTS_NUM)

  CreatePointsChild - Adds points data to pointsItems vector for a QTreeWidget.
- int getShapeId () const

  Accessor Functions - Returns the data named after them.
- int getTrackerId () const
- string getShapeType () const
- bool getSelected () const
- int getX () const
- int getY () const
- QPainter & getPainter ()
- QTreeWidgetItem * getParentItem ()
- alpha::vector< QTreeWidgetItem * > & getChildItems ()
- alpha::vector< QTreeWidgetItem * > & getPointsItems ()
- alpha::vector< QTreeWidgetItem * > & getPenItems ()
- alpha::vector< QTreeWidgetItem * > & getBrushItems ()
- int getPenWidth () const
- PenStyle getPenStyle () const

- PenCapStyle getPenCapStyle () const
- PenJoinStyle getPenJoinStyle () const
- QColor getPenColor () const
- QColor getBrushColor () const
- BrushStyle getBrushStyle () const
- QPen getPen () const
- QBrush getBrush () const
- QPoint getPoints () const
- int getChildEnd () const
- int getPenItemsEnd () const
- int getBrushItemsEnd () const
- void setShapeId (int shapeId)

  Accessor Functions.
- void setShapeType (string shapeType)
- void setSelected (bool selected)
- void setTrackerId (int trackerId)
- void allocateTrackerId (int shapeId)
- void setPen (GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle, PenJoin↩ Style penJoinStyle)
- void setBrush (GlobalColor brushColor, BrushStyle brushStyle)
- QPen & setInternalPen ()
- QBrush & setInternalBrush ()

Additional Inherited Members

Static Public Attributes inherited from Shape

- static int nextTracker [9] = {}
- static bool trackersInUse [9000] = {}

Protected Attributes inherited from Shape

- QTreeWidgetItem ∗ parentItem

  Mutator Functions.
- alpha::vector< QTreeWidgetItem ∗ > childItems

  vector of QTreeWidgetItem∗ holding data of all child items in parentItem
- alpha::vector< QTreeWidgetItem ∗ > pointsItems

  vector of QTreeWidgetItem∗ holding data of all points (besides coords) in parentItem
- alpha::vector< QTreeWidgetItem ∗ > penItems

  vector of QTreeWidgetItem∗ holding data of all pen items
- alpha::vector< QTreeWidgetItem ∗ > brushItems

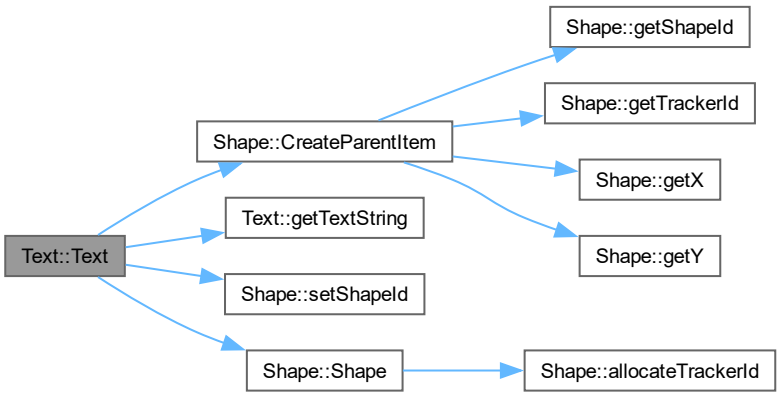
  vector of QTreeWidgetItem∗ holding data of all brush items

8.24.1 Constructor & Destructor Documentation

8.24.1.1 Text()

```
Text::Text (
            string shapeType,
            QPoint coords,
            QString textString,
            GlobalColor textColor,
            AlignmentFlag textAlignment,
            QFont font,
            int length,
            int width)
```

Here is the call graph for this function:



8.24.2 Member Function Documentation

8.24.2.1 Area()

```
double Text::Area () const  [override], [virtual]
```

Area - Returns the area of the shape.

Returns

Implements Shape.

8.24.2.2 Draw()

```
void Text::Draw (
            QWidget * renderArea)  [override], [virtual]
```

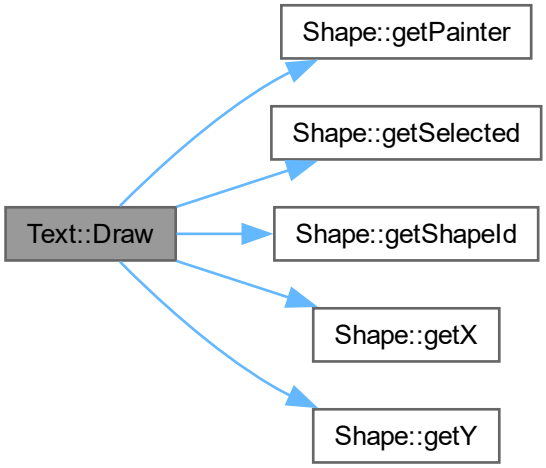
Draw - Draws the shape to the associated renderArea.

Parameters

renderArea	- QWidget to be drawn on

Implements Shape.

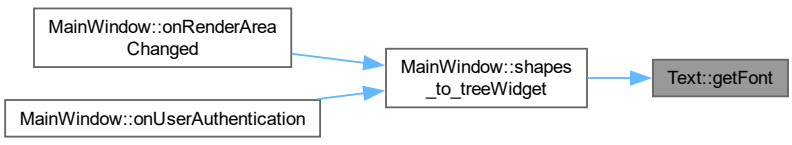
Here is the call graph for this function:



8.24.2.3 getFont()

```
QFont Text::getFont () const
```

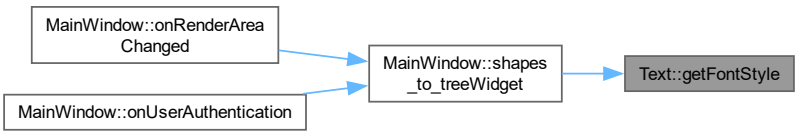
Here is the caller graph for this function:



8.24.2.4 getFontStyle()

```
int Text::getFontStyle () const
```

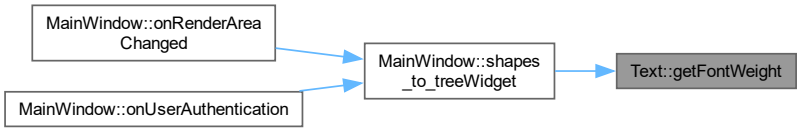
Here is the caller graph for this function:



8.24.2.5 getFontWeight()

```
QFont::Weight Text::getFontWeight () const
```

Here is the caller graph for this function:



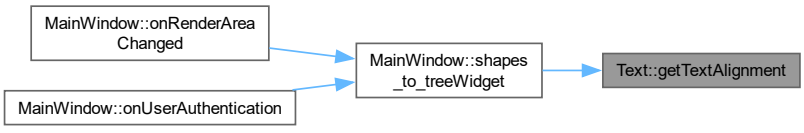
8.24.2.6 getLength()

```
int Text::getLength () const
```

8.24.2.7 getTextAlignment()

```
AlignmentFlag Text::getTextAlignment () const
```

Here is the caller graph for this function:



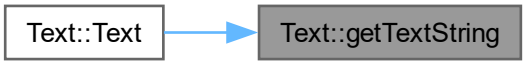
8.24.2.8 getTextColor()

```
GlobalColor Text::getTextColor () const
```

8.24.2.9 getTextString()

```
QString Text::getTextString () const
```

Here is the caller graph for this function:



8.24.2.10 getWidth()

```
int Text::getWidth () const
```

8.24.2.11 isPointInside()

```
bool Text::isPointInside (
            const QPoint & point) const  [override], [virtual]
```

isPointInside - Returns true if point is inside the shape

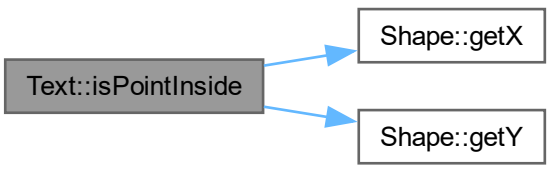
Parameters

point	- QPoint being checked

Returns

Implements Shape.

Here is the call graph for this function:



8.24.2.12 Perimeter()

```
double Text::Perimeter () const  [override], [virtual]
```

Perimeter - Returns the perimeter of the shape.

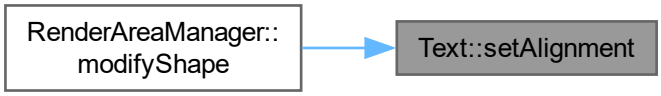
Returns

Implements Shape.

8.24.2.13 setAlignment()

```
void Text::setAlignment (
            Qt::AlignmentFlag alignment)
```

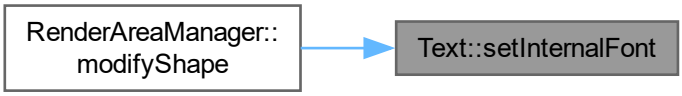
Here is the caller graph for this function:



8.24.2.14 setInternalFont()

```
QFont & Text::setInternalFont ()
```

Here is the caller graph for this function:



8.24.2.15 setLength()

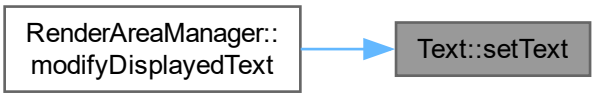
```
void Text::setLength (
            int newLength)
```

Here is the caller graph for this function:

8.24.2.16 setText()

```
void Text::setText (
            QString text)
```

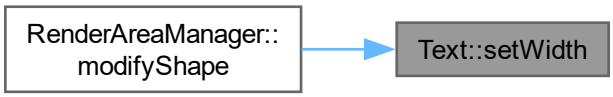
Here is the caller graph for this function:



8.24.2.17 setWidth()

```
void Text::setWidth (
            int newWidth)
```

Here is the caller graph for this function:

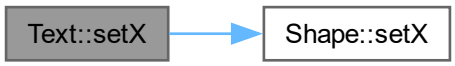


8.24.2.18 setX()

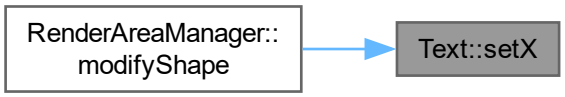
```
void Text::setX (
            int newX)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:

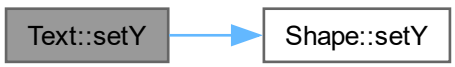


8.24.2.19 setY()

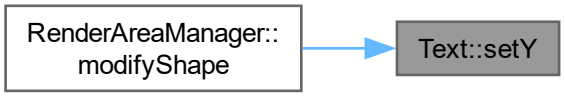
```
void Text::setY (
            int newY)
```

Implements Shape.

Here is the call graph for this function:



Here is the caller graph for this function:



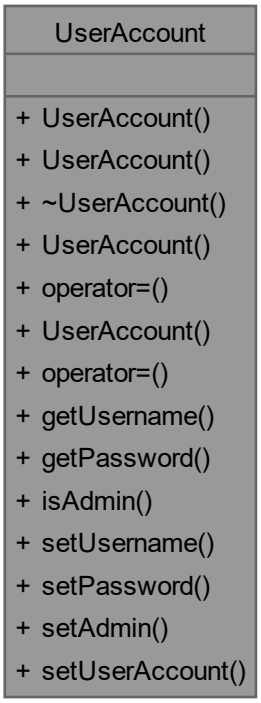
The documentation for this class was generated from the following files:

- src/code/text.h
- src/code/text.cpp

8.25 UserAccount Class Reference

`#include <UserAccount.h>`

Collaboration diagram for UserAccount:



Public Member Functions

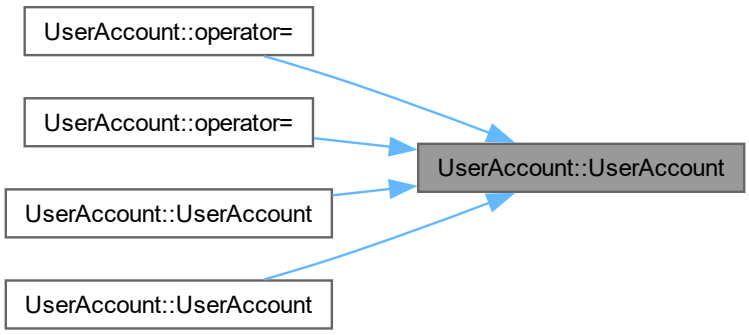
- UserAccount ()
- UserAccount (QString username, QString password, bool admin)
- ~UserAccount ()
- UserAccount (const UserAccount &other)
- UserAccount & operator= (const UserAccount &other)
- UserAccount (UserAccount &&other) noexcept
- UserAccount & operator= (UserAccount &&other) noexcept
- QString getUsername () const
- QString getPassword () const
- bool isAdmin () const
- void setUsername (QString &username)
- void setPassword (QString &password)
- void setAdmin (bool admin)
- void setUserAccount (QString &username, QString &password, bool admin)

8.25.1 Constructor & Destructor Documentation

8.25.1.1 UserAccount() [1/4]

```
UserAccount::UserAccount ()
```

Here is the caller graph for this function:



8.25.1.2 UserAccount() [2/4]

```
UserAccount::UserAccount (
            QString username,
            QString password,
            bool admin)
```

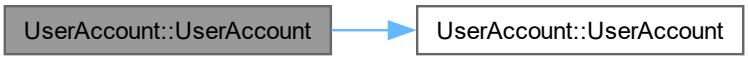
8.25.1.3 ~UserAccount()

```
UserAccount::~UserAccount ()
```

8.25.1.4 UserAccount() [3/4]

```
UserAccount::UserAccount (
            const UserAccount & other)
```

Here is the call graph for this function:



8.25.1.5 UserAccount() [4/4]

```
UserAccount::UserAccount (
            UserAccount && other)  [noexcept]
```

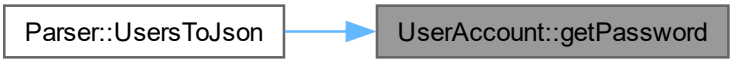
Here is the call graph for this function:

8.25.2 Member Function Documentation

8.25.2.1 getPassword()

```
QString UserAccount::getPassword () const
```

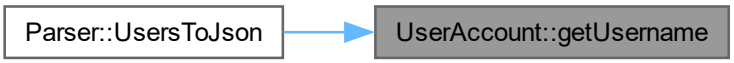
Here is the caller graph for this function:



8.25.2.2 getUsername()

```
QString UserAccount::getUsername () const
```

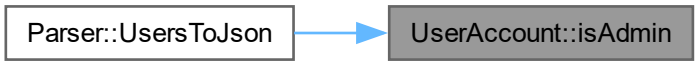
Here is the caller graph for this function:



8.25.2.3 isAdmin()

```
bool UserAccount::isAdmin () const
```

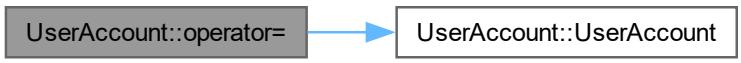
Here is the caller graph for this function:



8.25.2.4 operator=() [1/2]

```
UserAccount & UserAccount::operator= (
            const UserAccount & other)
```

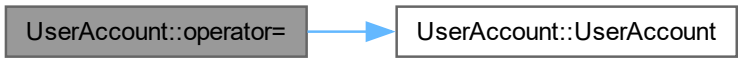
Here is the call graph for this function:



8.25.2.5 operator=() [2/2]

```
UserAccount & UserAccount::operator= (
            UserAccount && other)  [noexcept]
```

Here is the call graph for this function:



8.25.2.6 setAdmin()

```
void UserAccount::setAdmin (
            bool admin)
```

8.25.2.7 setPassword()

```
void UserAccount::setPassword (
            QString & password)
```

8.25.2.8 setUserAccount()

```
void UserAccount::setUserAccount (
            QString & username,
            QString & password,
            bool admin)
```

8.25.2.9 setUsername()

```
void UserAccount::setUsername (
            QString & username)
```

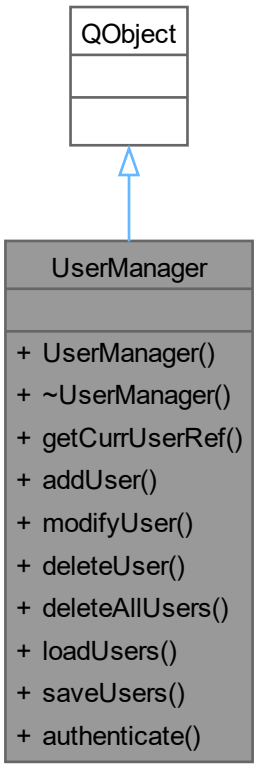
The documentation for this class was generated from the following files:

- src/code/UserAccount.h
- src/code/UserAccount.cpp

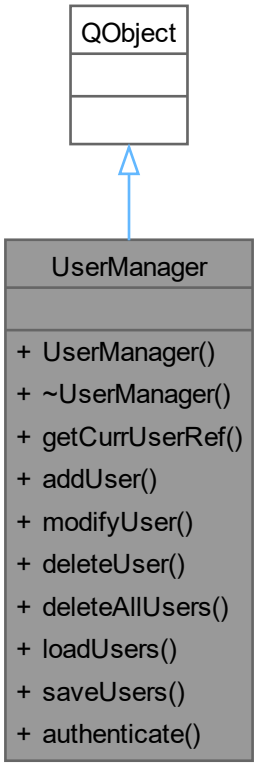
8.26 UserManager Class Reference

`#include <UserManager.h>`

Inheritance diagram for UserManager:



Collaboration diagram for UserManager:



Signals

- void userChanged ()
- void userNotChanged (const QString &message)
- void statusMessage (const QString &message)
- void userAuthenticated (const UserAccount ∗currUser)
- void authenticationFailed (const QString &message)

Public Member Functions

- UserManager (QObject ∗parent=nullptr)

  Default Constructor.
- ∼UserManager ()
- UserAccount ∗ getCurrUserRef ()

  Returns the users vector as a reference.
- void addUser (const QString username, const QString password, const bool admin)

  These functions are used to add, change, delete, and load users in the UserManager.
- void modifyUser (const QString username, const QString password, const bool admin)
- void deleteUser (QString username)
- void deleteAllUsers ()
- void loadUsers ()
- void saveUsers ()
- void authenticate (const QString username, const QString password)

8.26.1 Constructor & Destructor Documentation

8.26.1.1 UserManager()

```
UserManager::UserManager (
            QObject * parent = nullptr)  [explicit]
```

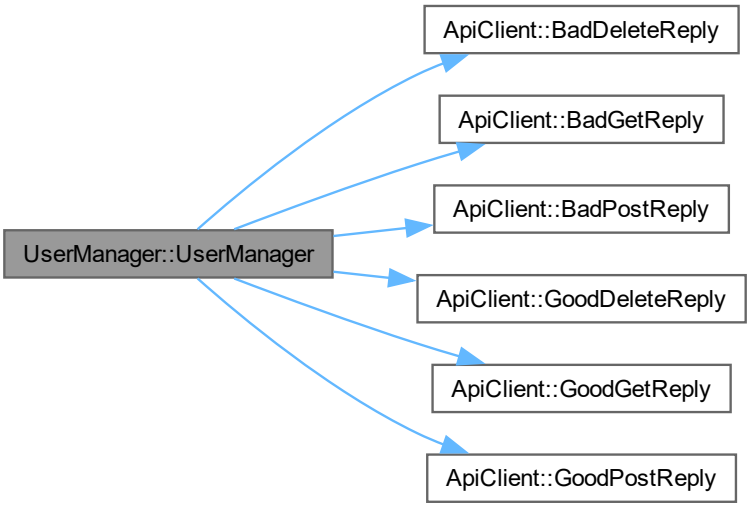
Default Constructor.

This constructor initializes the UserManager object and connects to the ApiClient signals for handling user data.

Parameters

parent	- any QObject to tie this instantiation to ensure automatic deletion

Here is the call graph for this function:



8.26.1.2 ~UserManager()

```
UserManager::~UserManager ()
```

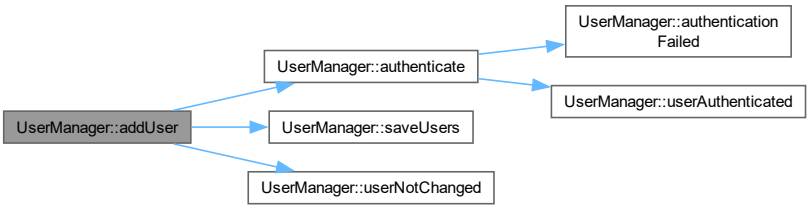
8.26.2 Member Function Documentation

8.26.2.1 addUser()

```
void UserManager::addUser (
            const QString username,
            const QString password,
            const bool admin)
```

These functions are used to add, change, delete, and load users in the UserManager.

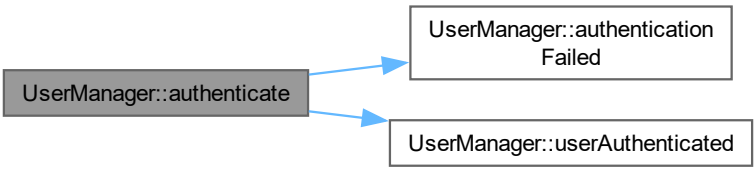
All of these functions emit the userChanged() signal to notify the frontend that the user list has changed and needs to be refreshed except for authenticate(). Here is the call graph for this function:



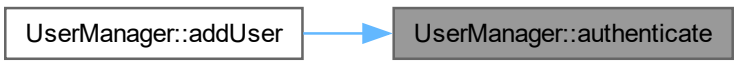
8.26.2.2 authenticate()

```
void UserManager::authenticate (
            const QString username,
            const QString password)
```

Here is the call graph for this function:



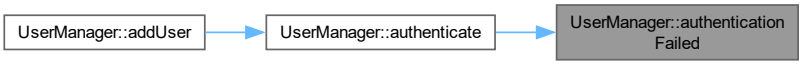
Here is the caller graph for this function:



8.26.2.3 authenticationFailed

```
void UserManager::authenticationFailed (
            const QString & message)  [signal]
```

Here is the caller graph for this function:



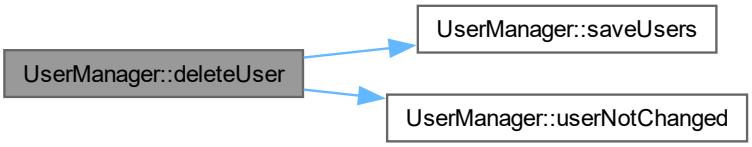
8.26.2.4 deleteAllUsers()

```
void UserManager::deleteAllUsers ()
```

8.26.2.5 deleteUser()

```
void UserManager::deleteUser (
            QString username)
```

Here is the call graph for this function:



8.26.2.6 getCurrUserRef()

```
UserAccount * UserManager::getCurrUserRef ()
```

Returns the users vector as a reference.

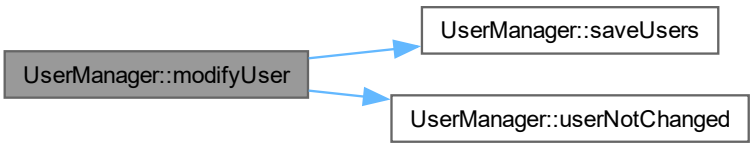
8.26.2.7 loadUsers()

```
void UserManager::loadUsers ()
```

8.26.2.8 modifyUser()

```
void UserManager::modifyUser (
            const QString username,
            const QString password,
            const bool admin)
```

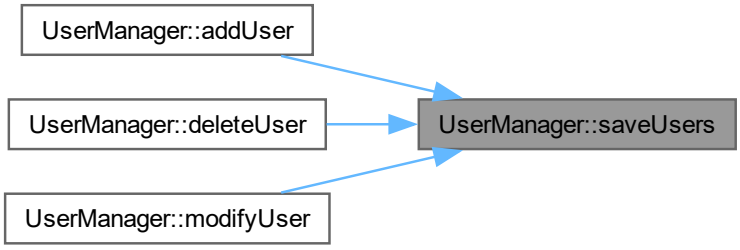
Here is the call graph for this function:



8.26.2.9 saveUsers()

```
void UserManager::saveUsers ()
```

Here is the caller graph for this function:



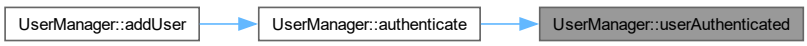
8.26.2.10 statusMessage

```
void UserManager::statusMessage (
            const QString & message)  [signal]
```

8.26.2.11 userAuthenticated

```
void UserManager::userAuthenticated (
            const UserAccount * currUser)  [signal]
```

Here is the caller graph for this function:



8.26.2.12 userChanged

```
void UserManager::userChanged ()  [signal]
```

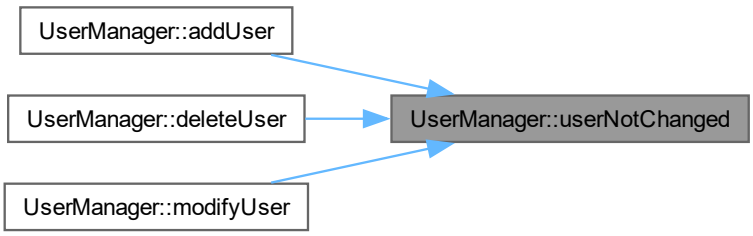
These signals are meant to connect to the User window in the frontend.

- userChanged(): signal for when the user list changes so the UI can refresh
- statusMessage(): passes status updates (e.g., save/load success or errors)
- userAuthenticated(): notifies when authentication succeeds or fails

8.26.2.13 userNotChanged

```
void UserManager::userNotChanged (
            const QString & message)  [signal]
```

Here is the caller graph for this function:



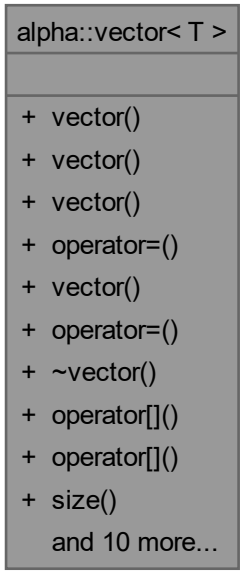
The documentation for this class was generated from the following files:

- src/code/UserManager.h
- src/code/moc_UserManager.cpp
- src/code/UserManager.cpp

8.27 alpha::vector< T > Class Template Reference

`#include <vector.h>`

Collaboration diagram for alpha::vector< T >:



Public Types

- using iterator = T*
- using const_iterator = const T*

Public Member Functions

- vector ()
- vector (int s)
- vector (const vector &other)
- vector & operator= (const vector &other)
- vector (vector &&other) noexcept
- vector & operator= (vector &&other) noexcept
- ~vector ()
- T & operator[ ] (int n)
- const T & operator[ ] (int n) const
- int size () const
- int capacity () const
- void resize (int newsize)
- void push_back (const T val)
- void reserve (int newalloc)
- iterator begin ()
- const_iterator begin () const
- iterator end ()
- const_iterator end () const
- iterator insert (iterator p, const T &v)
- iterator erase (iterator p)

8.27.1 Member Typedef Documentation

8.27.1.1 const_iterator

```
template<class T>
using alpha::vector< T >::const_iterator = const T*
```

8.27.1.2 iterator

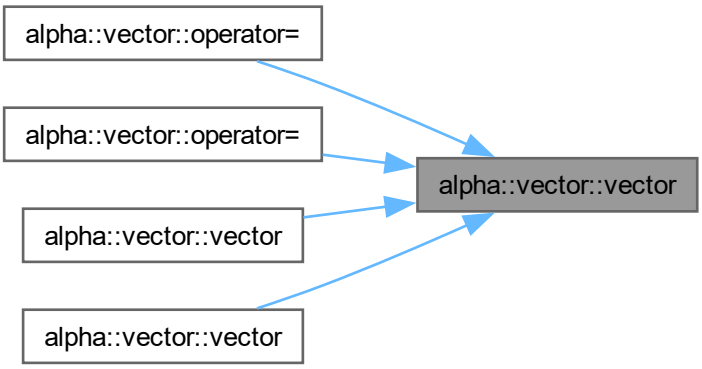
```
template<class T>
using alpha::vector< T >::iterator = T*
```

8.27.2 Constructor & Destructor Documentation

8.27.2.1 vector() [1/4]

```
template<class T>
alpha::vector< T >::vector ()  [inline]
```

Here is the caller graph for this function:



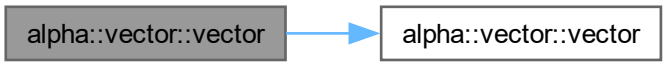
8.27.2.2 vector() [2/4]

```
template<class T>
alpha::vector< T >::vector (
            int s)  [inline], [explicit]
```

8.27.2.3 vector() [3/4]

```
template<class T>
alpha::vector< T >::vector (
            const vector< T > & other)  [inline]
```

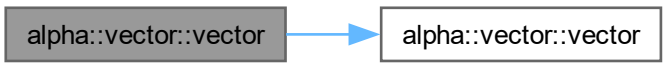
Here is the call graph for this function:



8.27.2.4 vector() [4/4]

```
template<class T>
alpha::vector< T >::vector (
            vector< T > && other)  [inline], [noexcept]
```

Here is the call graph for this function:



8.27.2.5 ~vector()

```
template<class T>
alpha::vector< T >::~vector ()  [inline]
```

8.27.3 Member Function Documentation

8.27.3.1 begin() [1/2]

```
template<class T>
iterator alpha::vector< T >::begin ()  [inline]
```

8.27.3.2 begin() [2/2]

```
template<class T>
const_iterator alpha::vector< T >::begin () const  [inline]
```

8.27.3.3 capacity()

```
template<class T>
int alpha::vector< T >::capacity () const  [inline]
```

8.27.3.4 end() [1/2]

```
template<class T>
iterator alpha::vector< T >::end ()  [inline]
```

Here is the caller graph for this function:

8.27.3.5 end() [2/2]

```
template<class T>
const_iterator alpha::vector< T >::end () const  [inline]
```

8.27.3.6 erase()

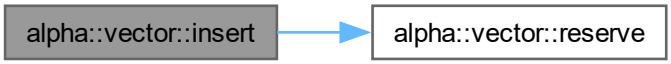
```
template<class T>
iterator alpha::vector< T >::erase (
            iterator p)  [inline]
```

Here is the call graph for this function:

8.27.3.7 insert()

```
template<class T>
iterator alpha::vector< T >::insert (
            iterator p,
            const T & v)  [inline]
```

Here is the call graph for this function:



8.27.3.8 operator=() [1/2]

```
template<class T>
vector & alpha::vector< T >::operator= (
            const vector< T > & other)  [inline]
```

Here is the call graph for this function:

8.27.3.9 operator=() [2/2]

```
template<class T>
vector & alpha::vector< T >::operator= (
            vector< T > && other)  [inline], [noexcept]
```

Here is the call graph for this function:

8.27.3.10 operator[]() [1/2]

```
template<class T>
T & alpha::vector< T >::operator[] (
            int n)  [inline]
```

8.27.3.11 operator[]() [2/2]

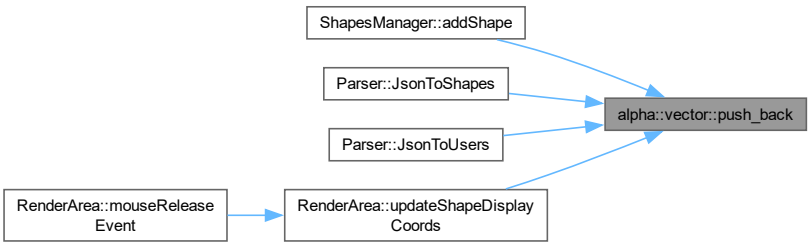
```
template<class T>
const T & alpha::vector< T >::operator[] (
            int n) const  [inline]
```

8.27.3.12 push_back()

```
template<class T>
void alpha::vector< T >::push_back (
            const T val)  [inline]
```

Here is the call graph for this function:

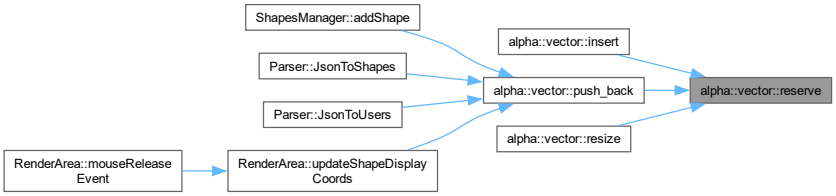
Here is the caller graph for this function:



8.27.3.13 reserve()

```
template<class T>
void alpha::vector< T >::reserve (
            int newalloc)  [inline]
```

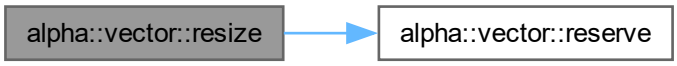
Here is the caller graph for this function:



8.27.3.14 resize()

```
template<class T>
void alpha::vector< T >::resize (
            int newsize)  [inline]
```

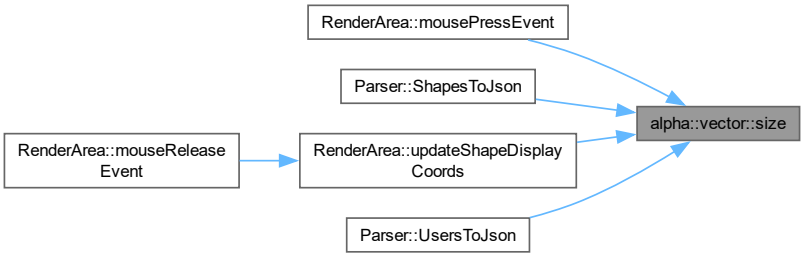
Here is the call graph for this function:



8.27.3.15 size()

```
template<class T>
int alpha::vector< T >::size () const  [inline]
```

Here is the caller graph for this function:



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

- src/code/vector.h

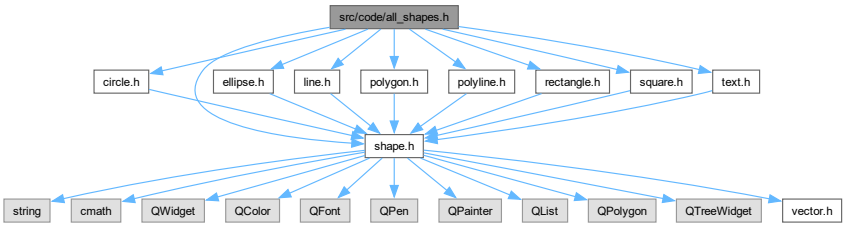
Chapter 9

File Documentation

9.1 src/code/all_shapes.h File Reference

```
#include "circle.h"
#include "ellipse.h"
#include "line.h"
#include "polygon.h"
#include "polyline.h"
#include "rectangle.h"
#include "shape.h"
#include "square.h"
#include "text.h"
```

Include dependency graph for all_shapes.h:



Enumerations

- enum ShapeIDs {
  LINE = 1 , POLYLINE , POLYGON , RECTANGLE ,
  SQUARE , ELLIPSE , CIRCLE , TEXT }

9.1.1 Enumeration Type Documentation

9.1.1.1 ShapeIDs

```
enum ShapeIDs
```

Enumerator

LINE	
POLYLINE	
POLYGON	
RECTANGLE	
SQUARE	
ELLIPSE	
CIRCLE	
TEXT	

9.2 all_shapes.h

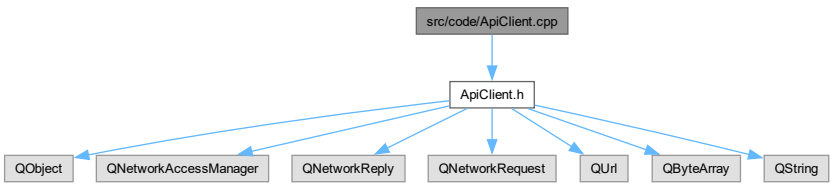
Go to the documentation of this file.

```
00001 // all_shapes.h | Include this file to automatically include all of these shapes in one line
00002 #ifndef ALL_SHAPES_H
00003 #define ALL_SHAPES_H
00004
00005 #include "circle.h"
00006 #include "ellipse.h"
00007 #include "line.h"
00008 #include "polygon.h"
00009 #include "polyline.h"
00010 #include "rectangle.h"
00011 #include "shape.h"
00012 #include "square.h"
00013 #include "text.h"
00014
00015 enum ShapeIDs {LINE = 1, POLYLINE, POLYGON, RECTANGLE, SQUARE, ELLIPSE, CIRCLE, TEXT};
00016
00017 #endif // ALL_SHAPES_H
```

9.3 src/code/ApiClient.cpp File Reference

```
#include "ApiClient.h"
```

Include dependency graph for ApiClient.cpp:



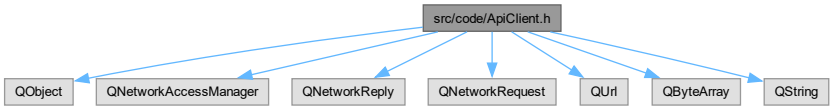
9.4 src/code/ApiClient.h File Reference

Implements the ApiClient class that makes API requests to the Crow Webservice.

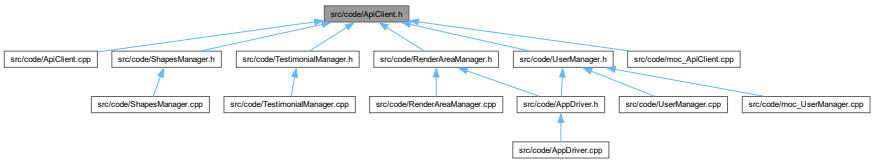
```
#include <QObject>
#include <QNetworkAccessManager>
```

```
#include <QNetworkReply>
#include <QNetworkRequest>
#include <QUrl>
#include <QByteArray>
#include <QString>
```

Include dependency graph for ApiClient.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ApiClient

Documentation Qt Version 6.9.0.

9.4.1 Detailed Description

Implements the ApiClient class that makes API requests to the Crow Webservice.

This class interacts with the following endpoints through the corresponding member functions Get /shapes : Get↩ Shapes() Post /shapes : PostShapes() Get /render_area : GetRenderArea() Post /render_area : PostRenderArea()

9.5 ApiClient.h

Go to the documentation of this file.

```
00001
00011 #ifndef API_CLIENT_H
00012 #define API_CLIENT_H
00014 #include <QObject>                 // https://doc.qt.io/qt-6/qobject.html
00015 #include <QNetworkAccessManager>    // https://doc.qt.io/qt-6/qnetworkaccessmanager.html
00016 #include <QNetworkReply>           // https://doc.qt.io/qt-6/qnetworkreply.html
00017 #include <QNetworkRequest>         // https://doc.qt.io/qt-6/qnetworkrequest.html
00018 #include <QUrl>                    // https://doc.qt.io/qt-6/qurl.html
00019 #include <QByteArray>              // https://doc.qt.io/qt-6/qbytearray.html
00020 #include <QString>                 // https://doc.qt.io/qt-6/qstring.html
00021
00022 class ApiClient : public QObject {
00023     Q_OBJECT    //necessary macro for qt's compiler
00024
00025 public:
```

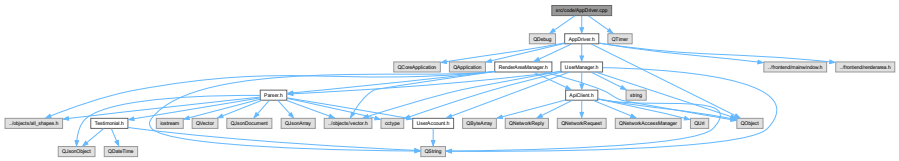
```
00031     explicit ApiClient(QObject* parent = nullptr);
00032
00033
00034
00042     void GetShapes();
00043
00044
00045
00053     void GetRenderArea();
00054
00055
00056
00063     void GetTestimonials();
00064
00065
00066
00074     void GetUsers();
00075
00076
00077
00088     void PostShapes(std::string json);
00089
00090
00091
00102     void PostRenderArea(std::string json);
00103
00104
00115     void PostUsers(std::string json);
00116
00117
00118
00127     void PostTestimonials(std::string json);
00128
00129
00130
00138     void DeleteShapesAll();
00139
00140
00141
00149     void DeleteRenderAreaAll();
00150
00151
00152
00160     void DeleteUsersAll();
00161
00162
00163
00170     void DeleteTestimonialsAll();
00171
00172
00173
00174 signals:
00183     void GoodGetReply(const QString& json);
00184
00185
00186
00195     void BadGetReply(const QString& error);
00196
00197
00198
00204     void GoodPostReply();
00205
00206
00207
00216     void BadPostReply(const QString& error);
00217
00218
00219
00225     void GoodDeleteReply();
00226
00227
00228
00237     void BadDeleteReply(const QString& error);
00238
00239
00240
00241 private slots:
00248     void AnalyzeGetReply();
00249
00250
00251
00258     void AnalyzePostReply();
00259
00260
00261
00268     void AnalyzeDeleteReply();
00269
```

```
00270
00271
00272 private:
00273     QNetworkAccessManager* manager;  // Manages all network operations for the client
00274 };
00275
00276 #endif //API_CLIENT_H
```

9.6 src/code/AppDriver.cpp File Reference

```
#include <QDebug>
#include "AppDriver.h"
#include <QTimer>
```

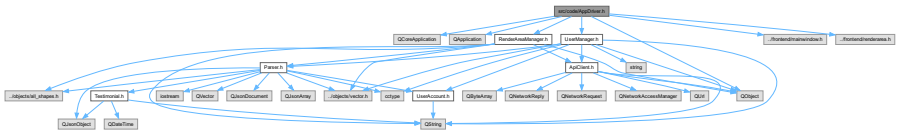
Include dependency graph for AppDriver.cpp:



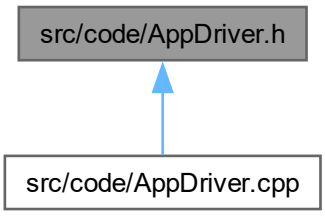
9.7 src/code/AppDriver.h File Reference

```
#include <QCoreApplication>
#include <QApplication>
#include <QObject>
#include "RenderAreaManager.h"
#include "UserManager.h"
#include "../frontend/mainwindow.h"
#include "../frontend/renderarea.h"
```

Include dependency graph for AppDriver.h:



This graph shows which files directly or indirectly include this file:



Classes

- class AppDriver

9.8 AppDriver.h

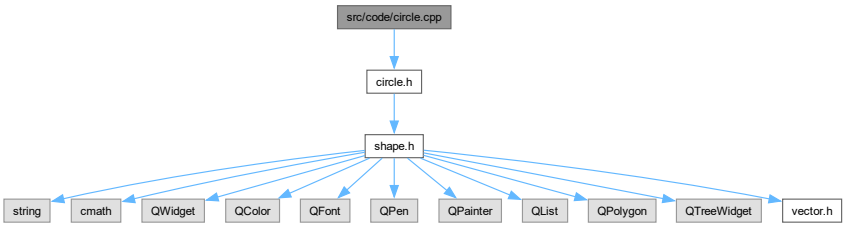
Go to the documentation of this file.

```
00001 #ifndef APP_DRIVER_H
00002 #define APP_DRIVER_H
00003 #include <QCoreApplication>
00004 #include <QApplication>
00005 #include <QObject>
00006 #include "RenderAreaManager.h"
00007 #include "UserManager.h"
00008 #include "../frontend/mainwindow.h"
00009 #include "../frontend/renderarea.h"
00010
00011 class AppDriver : public QObject {
00012     Q_OBJECT
00013 public:
00014     AppDriver(QObject* parent = nullptr);
00015     ~AppDriver();
00016
00017     void run();
00018     void shutdown();
00019
00020     void loadAllData();
00021
00022
00023 private slots:
00025     // For when a new shape is added to the render area
00026     void onRenderShapeAdded(Shape* shape);
00027     // For when an existing shape is modified in the render area
00028     void onRenderShapeChanged(Shape* shape, QString key, int value);
00029     // For when a single shape is deleted from the render area
00030     void onRenderShapeDeleted(const int trackerId);
00031     // For when all shapes are deleted from the render area
00032     void onRenderDeleteAllShapes();
00033
00035     // For when a new user is created
00036     void onNewUser(const QString username, const QString password, const bool admin);
00037     // For when an existing user's login credentials are modified
00038     void onUserModified(const QString username, const QString password, const bool admin);
00039     // For when a single user is deleted
00040     void onUserDeleted(const QString username);
00041     // For when all users are deleted
00042     void onDeleteAllUsers();
00043     // For when a user attempts to login
00044     void onLoginAttempt(const QString username, const QString password);
00045
00046
00047 private:
00048     RenderAreaManager* renderedShapes;      // Manages all the shapes that are currently rendered
00049     UserManager* user;                      // Manages the current user and holds all existing users
      to authenticate against
00050
00051     MainWindow* mainWindow;
00052
00053     //Subroutines
00054     void connectFrontendToDriver();
00055     void connectManagersToFrontend();
00056 };
00057
00058 #endif // APP_DRIVER_H
```

9.9 src/code/circle.cpp File Reference

```
#include "circle.h"
```

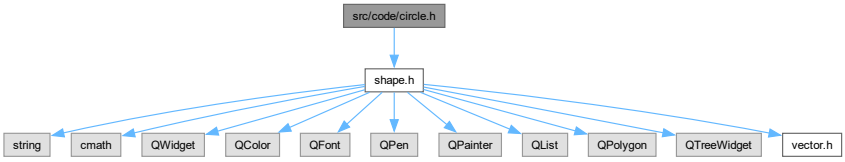
Include dependency graph for circle.cpp:



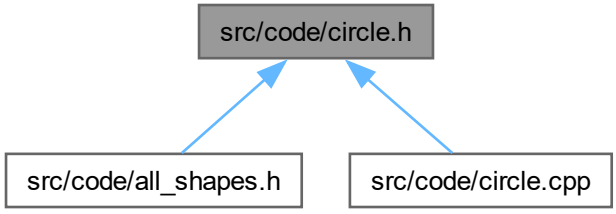
9.10 src/code/circle.h File Reference

```
#include "shape.h"
```

Include dependency graph for circle.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Circle

  The Circle class.

9.11 circle.h

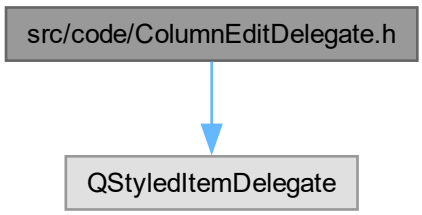
Go to the documentation of this file.

```
00001 #ifndef CIRCLE_H
00002 #define CIRCLE_H
00003
00004 #include "shape.h"
00005
00012
00013 class Circle : public Shape
00014 {
00015 public:
00024     Circle(string shapeType,
00025            QPoint coords,
00026            QPen   pen,
00027            QBrush brush,
00028            int r);
00029
00034     void Draw(QWidget* renderArea) override;
00035
00040     double Perimeter() const override;
00041
00046     double Area()      const override;
00047
00053     bool isPointInside(const QPoint& point) const override;
00054
00059     int getR() const;
00060
00062     void setR(int radius);
00063     void setX(int x);
00064     void setY(int y);
00066
00067 private:
00068     int r;
00069 };
00070
00071 #endif // CIRCLE_H
00072
```

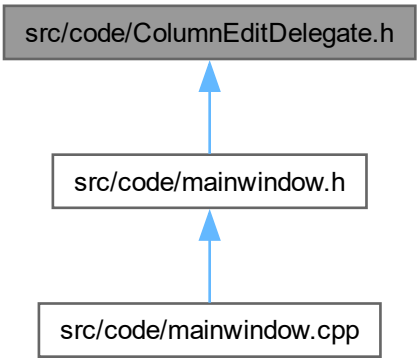
9.12 src/code/ColumnEditDelegate.h File Reference

`#include <QStyledItemDelegate>`

Include dependency graph for ColumnEditDelegate.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ColumnEditDelegate

9.13 ColumnEditDelegate.h

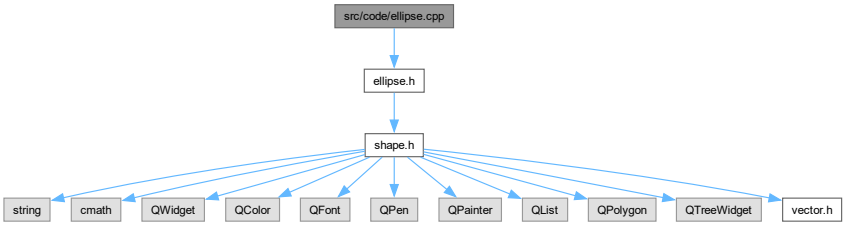
Go to the documentation of this file.

```
00001 #ifndef COLUMNEDITDELEGATE_H
00002 #define COLUMNEDITDELEGATE_H
00003
00004 #include <QStyledItemDelegate>
00005
00006 class ColumnEditDelegate : public QStyledItemDelegate
00007 {
00008 public:
00009     ColumnEditDelegate(QObject *parent = nullptr) : QStyledItemDelegate(parent) {}
00010
00011     QWidget *createEditor(QWidget *parent,
00012                           const QStyleOptionViewItem &option,
00013                           const QModelIndex &index) const override
00014     {
00015         if (index.column() != 1 || !canEdit)
00016         {
00017             return nullptr;  // Only allows editing in column 1
00018         }
00019
00020         return QStyledItemDelegate::createEditor(parent, option, index);
00021     }
00022
00023     void setCanEdit(bool edit)
00024     {
00025         canEdit = edit;
00026     }
00027
00028
00029 private:
00030     bool canEdit = false;  // Flag to control whether editing is allowed
00031 };
00032
00033 #endif // COLUMNEDITDELEGATE_H
```

9.14 src/code/ellipse.cpp File Reference

```
#include "ellipse.h"
```

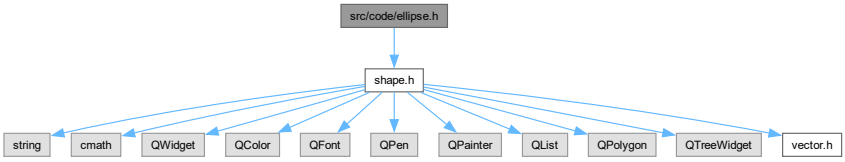
Include dependency graph for ellipse.cpp:



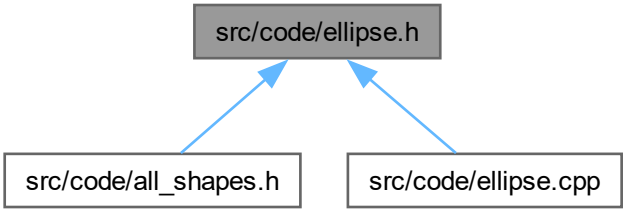
9.15 src/code/ellipse.h File Reference

```
#include "shape.h"
```

Include dependency graph for ellipse.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Ellipse

  The Ellipse class.

9.16 ellipse.h

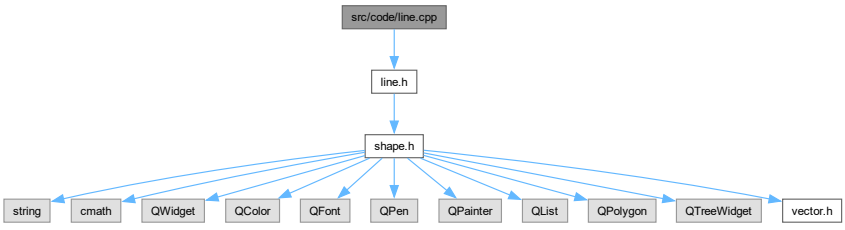
Go to the documentation of this file.

```
00001 #ifndef ELLIPSE_H
00002 #define ELLIPSE_H
00003
00004 #include "shape.h"
00005
00012 class Ellipse : public Shape
00013 {
00014 public:
00024     Ellipse(string shapeType,
00025             QPoint coords,
00026             QPen   pen,
00027             QBrush brush,
00028             int    a,
00029             int    b);
00030
00035     void Draw(QWidget* renderArea) override;
00036
00041     double Perimeter() const override;
00042
00047     double Area()      const override;
00048
00054     bool isPointInside(const QPoint& point) const override;
00055
00057     int getA() const;
00058     int getB() const;
00060
00062     void setA(int newA);
00063     void setB(int newB);
00064     void setX(int newX);
00065     void setY(int newY);
00067
00068 private:
00069     int a;
00070     int b;
00071 };
00072
00073 #endif // ELLIPSE_H
```

9.17 src/code/line.cpp File Reference

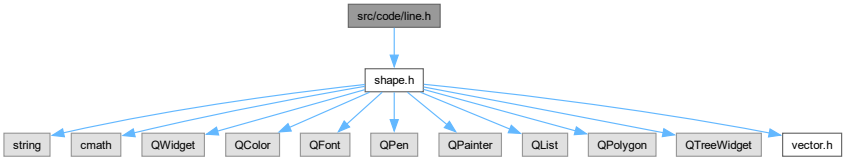
`#include "line.h"`

Include dependency graph for line.cpp:

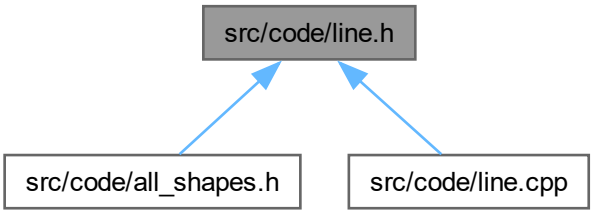


9.18 src/code/line.h File Reference

`#include "shape.h"`
Include dependency graph for line.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Line

  The Line class.

9.19 line.h

Go to the documentation of this file.

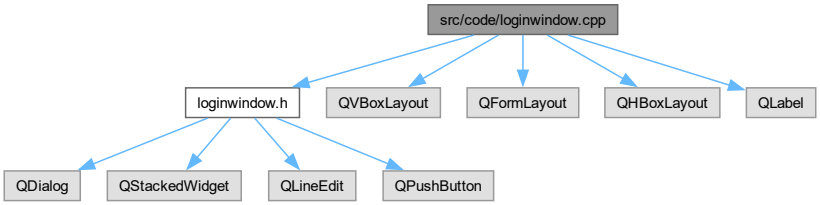
```
00001 #ifndef LINE_H
00002 #define LINE_H
00003
00004 #include "shape.h"
00005
00012 class Line : public Shape
00013 {
00014 public:
00024     Line(string shapeType,
00025          QPoint coords,
00026          QPen   pen,
00027          QBrush brush,
00028          QPoint startPoint,
00029          QPoint endPoint);
00030
00035     void Draw(QWidget* renderArea) override;
00036
00042     void Move(int x, int y) override;
00043
```

```
00048     double Perimeter() const override;
00049
00054     double Area() const override { return 0; }  // Need to implement this to instantiate Line
00055
00061     bool isPointInside(const QPoint& point) const override;
00062
00064     QPoint getStartPoint() const;
00065     QPoint getEndPoint()   const;
00067
00069     void setStartPoint(const QPoint& newStartPoint);
00070     void setEndPoint(const QPoint& newEndPoint);
00071     void setX(int newX);
00072     void setY(int newY);
00074
00075 private:
00076     QPoint startPoint;
00077     QPoint endPoint;
00078 };
00079
00080 #endif // LINE_H
```

9.20 src/code/loginwindow.cpp File Reference

```
#include "loginwindow.h"
#include <QVBoxLayout>
#include <QFormLayout>
#include <QHBoxLayout>
#include <QLabel>
```

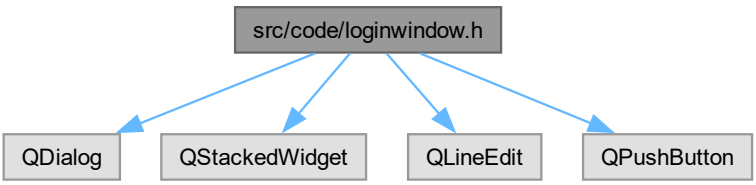
Include dependency graph for loginwindow.cpp:



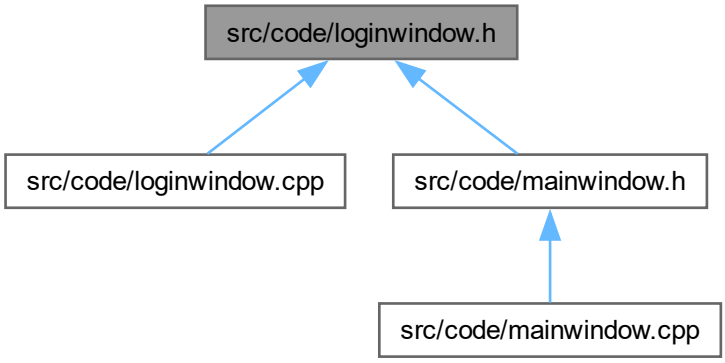
9.21 src/code/loginwindow.h File Reference

```
#include <QDialog>
#include <QStackedWidget>
#include <QLineEdit>
#include <QPushButton>
```

Include dependency graph for loginwindow.h:



This graph shows which files directly or indirectly include this file:



Classes

- class LoginWindow

9.22 loginwindow.h

Go to the documentation of this file.

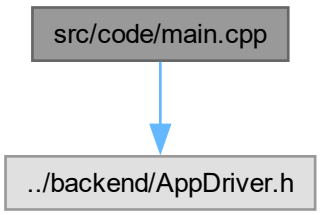
```
00001 #ifndef LOGIN_WINDOW_H
00002 #define LOGIN_WINDOW_H
00003
00004 #include <QDialog>
00005 #include <QStackedWidget>
00006 #include <QLineEdit>
00007 #include <QPushButton>
00008
00009 class LoginWindow : public QDialog {
00010     Q_OBJECT
00011
00012 public:
00013     explicit LoginWindow(QWidget *parent = nullptr);
00014
00015     QString username() const;
00016     QString password() const;
00017
00018 signals:
00019     void loginRequested(const QString &username, const QString &password);
00020     void signupRequested(const QString &username, const QString &password, const bool admin);
00021
00022 private slots:
00023     void showSignupPage();
00024     void showLoginPage();
00025     void attemptLogin();
00026     void attemptSignup();
00027
00028 private:
00029     QStackedWidget *stack;
00030
00031     // --- login page widgets
00032     QWidget *loginPage;
00033     QLineEdit *loginUserEdit;
00034     QLineEdit *loginPassEdit;
00035     QPushButton *loginBtn;
00036     QPushButton *toSignupBtn;
00037
00038     // --- signup page widgets
```

```
00039     QWidget *signupPage;
00040     QLineEdit *signupUserEdit;
00041     QLineEdit *signupPassEdit;
00042     QPushButton *signupBtn;
00043     QPushButton *backToLoginBtn;
00044 };
00045
00046 #endif // LOGINDIALOG_H
```

9.23 src/code/main.cpp File Reference

```
#include "../backend/AppDriver.h"
```

Include dependency graph for main.cpp:



Functions

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

9.23.1 Function Documentation

9.23.1.1 main()

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

Here is the call graph for this function:

9.24 src/code/mainwindow.cpp File Reference

```
#include "mainwindow.h"
#include "../backend/TestimonialManager.h"
#include "TestimonialDialog.h"
#include "TestimonialsDisplayDialog.h"
#include <QAction>
#include <QTimer>
#include <QMenuBar>
```

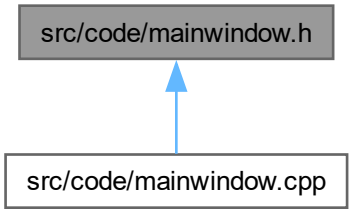
Include dependency graph for mainwindow.cpp:

9.25 src/code/mainwindow.h File Reference

```
#include <QMainWindow>
#include <algorithm>
#include <QTabWidget>
#include <QTableWidget>
#include <QComboBox>
#include <QApplication>
#include <QGridLayout>
#include <QLabel>
#include <QFile>
#include <QTimer>
#include <QStatusBar>
#include <QVBoxLayout>
#include <QPushButton>
#include <QSpinBox>
#include "ui_mainwindow.h"
#include "renderarea.h"
#include "loginwindow.h"
#include "ColumnEditDelegate.h"
#include "../backend/UserAccount.h"
```

Include dependency graph for mainwindow.h:

This graph shows which files directly or indirectly include this file:



Classes

- class MainWindow

Namespaces

- namespace Ui

9.26 mainwindow.h

Go to the documentation of this file.

```
00001 // mainwindow.h
00002 #ifndef MAINWINDOW_H
00003 #define MAINWINDOW_H
00004
00005 #include <QMainWindow>
00006 #include <algorithm>
00007 #include <QTabWidget>
00008 #include <QTableWidget>
00009 #include <QComboBox>
00010 #include <QApplication>
00011 #include <QGridLayout>
00012 #include <QLabel>
00013 #include <QFile>
00014 #include <QTimer>
00015 #include <QComboBox>
00016 #include <QStatusBar>
00017 #include <QVBoxLayout>
00018 #include <QPushButton>
00019 #include <QSpinBox>
00020 #include "ui_mainwindow.h"
00021 #include "renderarea.h"
00022 #include "loginwindow.h"
00023 #include "ColumnEditDelegate.h"
00024 #include "../backend/UserAccount.h"
00025
00026 QT_BEGIN_NAMESPACE
00027 namespace Ui {
00028 class MainWindow;
00029 }
00030 QT_END_NAMESPACE
00031
00032 class MainWindow : public QMainWindow
00033 {
00034     Q_OBJECT
00035
00036 public:
00037     explicit MainWindow(QWidget *parent = nullptr);
```

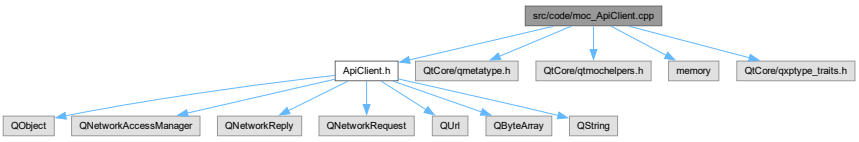
```
00038     MainWindow(QWidget *parent, const alpha::vector<Shape*>* renderedShapes, const UserAccount*
      currUser);
00039     ~MainWindow();
00040
00041     void drawShapes() const;
00042     void shapes_to_treeWidget();
00043
00044 signals:
00045     // shape signals
00046     void shapeAdded(Shape* shape);
00047     void shapeChanged(Shape* shape, QString key, int vaue);
00048     void displayedTextChanged(Text* text, QString newText);
00049     void shapeDeleted(int trackerId);
00050     void deleteAllShapes();
00051
00052     // login/signup signals into AppDriver
00053     void loginAttempt(const QString &username, const QString &password);
00054     void newUserAdded(const QString &username, const QString &password, const bool admin);
00055
00056     // dialog-control signals
00057     void loginSuccess();
00058     void loginFailed(const QString &message);
00059
00060 public slots:
00061     // Signals for these slots come from RenderAreaManager class
00062     void onRenderAreaChanged();
00063     void onRenderAreaNotChanged(const QString& message);
00064     void showRenderStatusMessage(const QString &message);
00065
00066     // login flow
00067     void onLoginClicked();
00068     void onLoginRequest(const QString &username, const QString &password);
00069     void onSignupRequest(const QString &username, const QString &password, const bool admin);
00070
00071     // responses from UserManager
00072     void onUserAuthentication(const UserAccount* currUser);
00073     void onUserAuthenticationFailure(const QString& message);
00074
00075 private slots:
00076
00077     void onToggleStyle(bool checked = true);
00078
00079     void on_actionnew_line_button_triggered();
00080     void on_actionnew_square_button_triggered();
00081     void on_actionnew_rectange_button_triggered();
00082     void on_actionnew_circle_button_triggered();
00083     void on_actionnew_ellipse_button_triggered();
00084     void on_actionnew_polyline_button_triggered();
00085     void on_actionnew_polygon_button_triggered();
00086     void on_actionnew_text_button_triggered();
00087     void on_actionremove_shape_button_triggered();
00088
00089     // For Selection Sort Algo
00090     void onSortMethodChanged(int index);
00091
00092     // For modifying shape
00093     void onTreeWidgetItemChanged(QTreeWidgetItem*, int column);
00094     void onComboBoxChanged(int newIndex);
00095     void onSpinBoxChanged();
00096
00097     void showTestimonialPrompt();
00098     void showTestimonialsDisplay();
00099
00100     // For contact us window
00101     void onContactUsClicked();
00102
00103 private:
00104     Ui::MainWindow *ui;
00105     RenderArea *renderArea;
00106
00107     const alpha::vector<Shape*>* renderShapes;      // Holds currently renderedShapes
00108     const UserAccount* currUser;
00109     QLabel *userStatusLabel;
00110
00111     void addToShapeTree(Shape* shape);
00112     ColumnEditDelegate* delegate;
00113
00114     QString loadStyleSheet(const QString &path);
00115
00116     QComboBox* createShapeTypeComboBox(const QString& currentShapeType);
00117
00118     QSpinBox*  createPenWidthSpinBox(int currentPenWidth);
00119     QComboBox* createColorComboBox(int currentColor);
00120     QComboBox* createPenStyleComboBox(int currentPenStyle);
00121     QComboBox* createPenCapStyleComboBox(int currentPenCapStyle);
00122     QComboBox* createPenJoinStyleComboBox(int currentPenJoinStyle);
00123
```

```
00124     QComboBox* createBrushStyleComboBox(int currentBrushStyle);
00125
00126     QComboBox* createAlignmentComboBox(AlignmentFlag currentAlignment);
00127     QComboBox* createFontComboBox(QFont currentFont);
00128     QComboBox* createFontStyleComboBox(int currentFontStyle);
00129     QComboBox* createFontWeightComboBox(QFont::Weight currentFontWeight);
00130
00131     void createShapeTableTab();
00132     QTabWidget* tabWidget; // Tab widget to manage tabs
00133     QTableWidget* shapeTable; // Table to display shapes
00134     QComboBox* sortDropdown; // Dropdown for sorting methods
00135     QComboBox* sortOrderDropdown; // Dropdown for sorting order
00136
00137     void selection_sort(alpha::vector<Shape*>& shapes, bool (*compare)(const Shape*, const Shape*),
      bool ascending);
00138
00139     void populateShapeTable(const alpha::vector<Shape*>& shapes);
00140
00141     static bool sortById(const Shape* a, const Shape* b);
00142     static bool sortByArea(const Shape* a, const Shape* b);
00143     static bool sortByPerimeter(const Shape* a, const Shape* b);
00144
00145     void setupTestimonials();
00146
00147     QWidget *contactUsWidget;
00148     QLabel *logoLabel;
00149     QLabel *teamNameLabel;
00150     QWidget *contactWindow;
00151 };
00152
00153 #endif // MAINWINDOW_H
```

9.27 src/code/moc_ApiClient.cpp File Reference

```
#include "ApiClient.h"
#include <QtCore/qmetatype.h>
#include <QtCore/qtmochelpers.h>
#include <memory>
#include <QtCore/qxptype_traits.h>
```

Include dependency graph for moc_ApiClient.cpp:



Classes

- struct QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t

Namespaces

- namespace QT_WARNING_DISABLE_DEPRECATED

Macros

- #define Q_CONSTINIT

9.27.1 Macro Definition Documentation

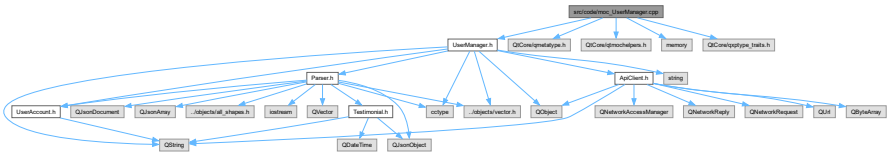
9.27.1.1 Q_CONSTINIT

```
#define Q_CONSTINIT
```

9.28 src/code/moc_UserManager.cpp File Reference

```
#include "UserManager.h"
#include <QtCore/qmetatype.h>
#include <QtCore/qtmochelpers.h>
#include <memory>
#include <QtCore/qxptype_traits.h>
```

Include dependency graph for moc_UserManager.cpp:



Classes

- struct QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t

Namespaces

- namespace QT_WARNING_DISABLE_DEPRECATED

Macros

- #define Q_CONSTINIT

9.28.1 Macro Definition Documentation

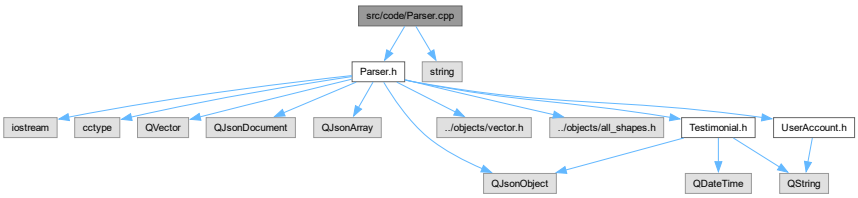
9.28.1.1 Q_CONSTINIT

```
#define Q_CONSTINIT
```

9.29 src/code/Parser.cpp File Reference

```
#include "Parser.h"
#include <string>
```

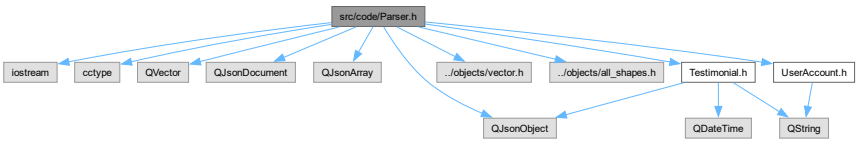
Include dependency graph for Parser.cpp:



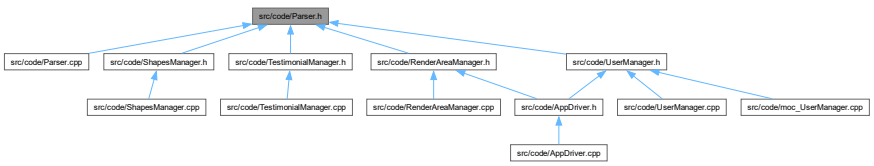
9.30 src/code/Parser.h File Reference

```
#include <iostream>
#include <cctype>
#include <QVector>
#include <QJsonDocument>
#include <QJsonArray>
#include <QJsonObject>
#include "../objects/vector.h"
#include "../objects/all_shapes.h"
#include "UserAccount.h"
#include "Testimonial.h"
```

Include dependency graph for Parser.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Parser

9.31 Parser.h

Go to the documentation of this file.

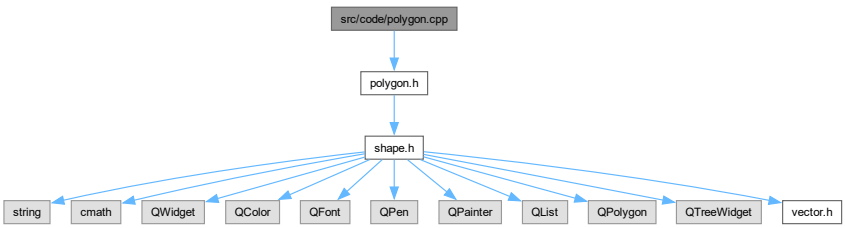
```
00001 #ifndef PARSER_H
00002 #define PARSER_H
00003
00004 #include <iostream>
00005 #include <cctype>
00006 #include <QVector>
00007 #include <QJsonDocument>
00008 #include <QJsonArray>
00009 #include <QJsonObject>
00010 #include "../objects/vector.h"
00011 #include "../objects/all_shapes.h"
00012 #include "UserAccount.h"
00013 #include "Testimonial.h"
00014
00015 class Parser {
00016 public:
00017     //Use default constructor/destructor as this class doesn't store any data
00018     Parser() = default;
00019     ~Parser() = default;
00020
00021     //Disable copying and moving
00022     Parser(const Parser&) = delete;
00023     Parser& operator=(const Parser&) = delete;
00024     Parser(Parser&&) = delete;
00025     Parser& operator=(Parser&&) = delete;
00026
00034     void PrintShapeVector(const alpha::vector<Shape*> &shapes);
00035
00045     alpha::vector<Shape*> JsonToShapes(const std::string& json);    //Forward Parser (JSON string ->
      vector<Shape*>)
00046
00057     std::string ShapesToJson(const alpha::vector<Shape*>& shapes);  //Reverse Parser (vector<Shape*>
      -> JSON string)
00058
00059     // Forward parser: JSON → vector<UserAccount>
00060     alpha::vector<UserAccount*> JsonToUsers(const std::string& json);
00061
00062     // Reverse parser: vector<UserAccount> → JSON
00063     std::string UsersToJson(const alpha::vector<UserAccount*>& users);
00064
00065     // Forward parser for testimonials
00066     static QVector<Testimonial> JsonToTestimonials(const std::string& json);
00067
00068     // Reverse parser for testimonials
00069     static std::string TestimonialsToJson(const QVector<Testimonial>& testimonials);
00070
00071 private:
00072     /*=================================== Forward Parser Subroutines
      ============================================*/
00083     struct MorphicShape {
00084         std::string shapeType = "";
00085         int shapeId = 0;
00086         int trackerId = 0;
00087         alpha::vector<int> shapeDimensions;
00088         QPen pen = QPen();
00089         QBrush brush = QBrush();
00090         QPoint coords = QPoint();
00091
00092         QString textString;
00093         GlobalColor textColor;
00094         QFont font;
00095         AlignmentFlag textAlignment;
00096     };
00097
00098     //Accumulator for user account objects
00099     struct RawUser {
00100             QString username;
00101             QString password;
00102             bool    admin       = false;
00103             bool    hasUsername = false;
00104             bool    hasPassword = false;
00105             bool    hasAdmin    = false;
00106         };
00107
00124     MorphicShape ParseJsonObject(const std::string json, size_t &index);
00125
00143     void UpdateAccumulator(const std::string &key, const std::string &value, MorphicShape &tempShape);
00144
00156     Shape* BuildShape(MorphicShape tempShape);
00157
00170     void SkipWhitespace(const std::string& json, size_t& index);
00171
```

```
00186     std::string ExtractKey(const std::string& json, size_t &index);
00187
00206     std::string ExtractValue(const std::string& json, size_t &index);
00207
00221     std::string ExtractInteger(const std::string& json, size_t &index);
00222
00237     std::string ExtractArray(const std::string& json, size_t &index);
00238
00239     std::string ExtractLiteral(const std::string& json, size_t& index);
00240
00253     alpha::vector<int> StringToVector(const std::string &value);
00254
00255     /*================================== Reverse Parser Subroutines
      ==========================================*/
00256
00278     std::string AppendCommonShapeData(const Shape* shape);
00279
00294     std::string AppendBrushData(const Shape* shape);
00295
00319     std::string AppendTextData(const Shape* shape);
00320
00333     std::string GetShapeDimensions(const Shape* shape);
00334
00345     std::string GetColor(const QColor &objectColor);
00346
00357     std::string GetPenStyle(const Shape* shape);
00358
00369     std::string GetPenCapStyle(const Shape* shape);
00370
00381     std::string GetPenJoinStyle(const Shape* shape);
00382
00393     std::string GetBrushStyle(const Shape* shape);
00394
00405     std::string GetAlignmentFlag(const Text* text);
00406
00417     std::string GetFontStyle(const Text* text);
00418
00429     std::string GetFontWeight(const Text* text);
00430
00431
00432
00433     static void   UpdateUserAccumulator(const std::string& key, const std::string& value, RawUser&
      acc);
00434 };
00435
00436 #endif // PARSER_H
```

9.32 src/code/polygon.cpp File Reference

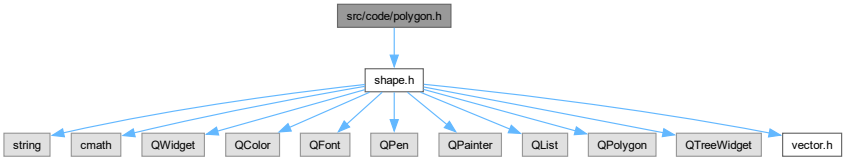
`#include "polygon.h"`

Include dependency graph for polygon.cpp:

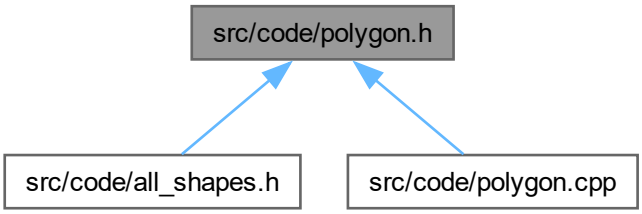


9.33 src/code/polygon.h File Reference

`#include "shape.h"`
Include dependency graph for polygon.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Polygon

  The Polygon class.

9.34 polygon.h

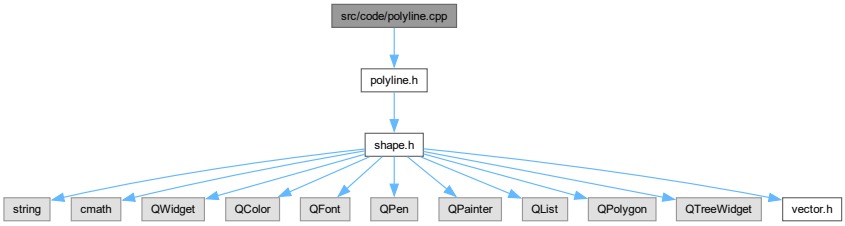
Go to the documentation of this file.

```
00001 #ifndef POLYGON_H
00002 #define POLYGON_H
00003
00004 #include "shape.h"
00005
00012 class Polygon : public Shape
00013 {
00014 public:
00023     Polygon(string shapeType,
00024             QPoint coords,
00025             QPen   pen,
00026             QBrush brush,
00027             QPolygon pointsList);
00028
00033     void Draw(QWidget* renderArea) override;
00034
00040     void Move(int x, int y) override;
00041
00046     double Perimeter() const override;
```

```
00047
00052     double Area() const override;
00053
00059     bool isPointInside(const QPoint& point) const override;
00060
00065     QPolygon getPointsList() const;
00066
00068
00072     void setPointsList(const QPolygon& newPointsList);
00073     void setX(int newX);
00074     void setY(int newY);
00076
00077 private:
00078     QPolygon pointsList;
00079 };
00080 #endif // POLYGON_H
```

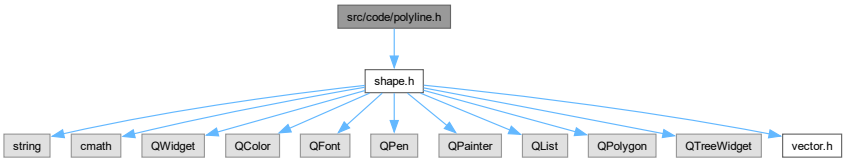
9.35 src/code/polyline.cpp File Reference

`#include "polyline.h"`
Include dependency graph for polyline.cpp:

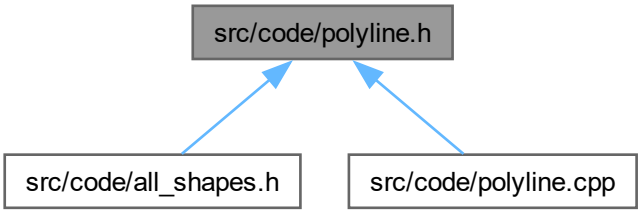


9.36 src/code/polyline.h File Reference

`#include "shape.h"`
Include dependency graph for polyline.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Polyline

9.37 polyline.h

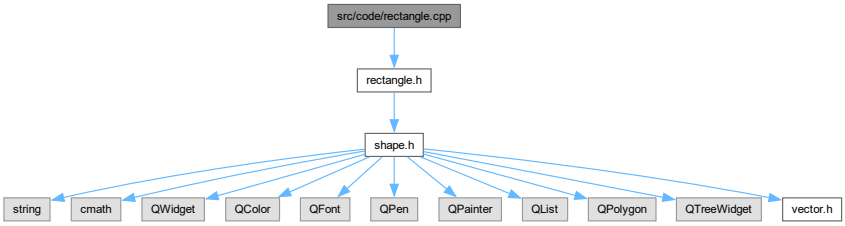
Go to the documentation of this file.

```
00001 #ifndef POLYLINE_H
00002 #define POLYLINE_H
00003
00004 #include "shape.h"
00005
00006 class Polyline : public Shape
00007 {
00008 public:
00009     Polyline(string shapeType,
00010              QPoint coords,
00011              QPen   pen,
00012              QBrush brush,
00013              QPolygon pointsList);
00014
00015     void Draw(QWidget* renderArea) override;
00016     void Move(int x, int y) override;
00017
00018     double Perimeter() const override;
00019     double Area() const override {return 0;}
00020
00021     bool isPointInside(const QPoint& point) const override;
00022
00023     QPolygon getPointsList() const;
00024
00025     void setPointsList(const QPolygon& newPointsList);
00026     void setX(int newX);
00027     void setY(int newY);
00028
00029 private:
00030     QPolygon pointsList;
00031 };
00032
00033 #endif // POLYLINE_H
```

9.38 src/code/rectangle.cpp File Reference

```
#include "rectangle.h"
```

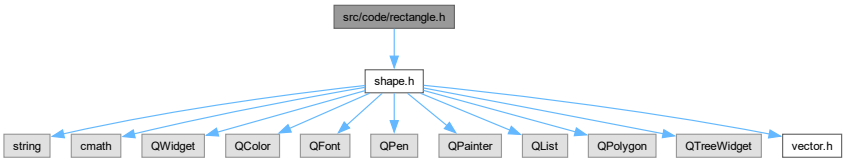
Include dependency graph for rectangle.cpp:



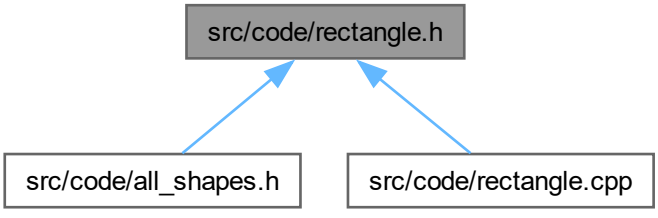
9.39 src/code/rectangle.h File Reference

```
#include "shape.h"
```

Include dependency graph for rectangle.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Rectangle

9.40 rectangle.h

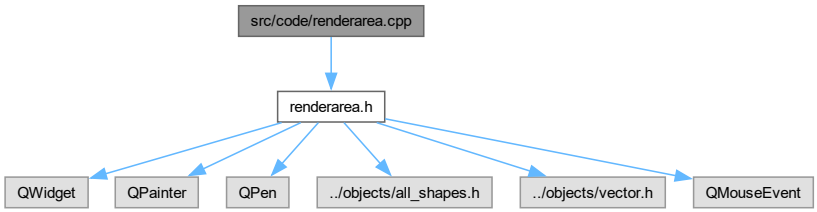
Go to the documentation of this file.

```
00001 #ifndef RECTANGLE_H
00002 #define RECTANGLE_H
00003
00004 #include "shape.h"
00005
00006 class Rectangle : public Shape
00007 {
00008 public:
00009     Rectangle(string shapeType,
00010               QPoint coords,
00011               QPen   pen,
00012               QBrush brush,
00013               int length,
00014               int width);
00015
00016
00017     void Draw(QWidget* renderArea) override;
00018
00019     double Perimeter() const override;
00020     double Area()      const override;
00021
00022     bool isPointInside(const QPoint& point) const override;
00023
00024     int getLength() const;
00025     int getWidth()  const;
00026
00027     void setLength(int newLength);
00028     void setWidth(int newWidth);
00029     void setX(int newX);
00030     void setY(int newY);
00031
00032 private:
00033     int length;
00034     int width;
00035 };
00036
00037 #endif // RECTANGLE_H
00038
```

9.41 src/code/renderarea.cpp File Reference

```
#include "renderarea.h"
```

Include dependency graph for renderarea.cpp:

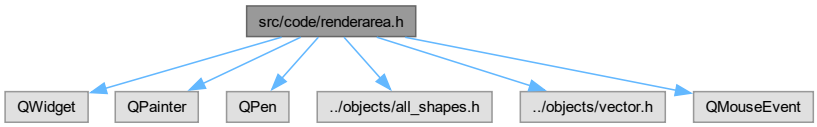


9.42 src/code/renderarea.h File Reference

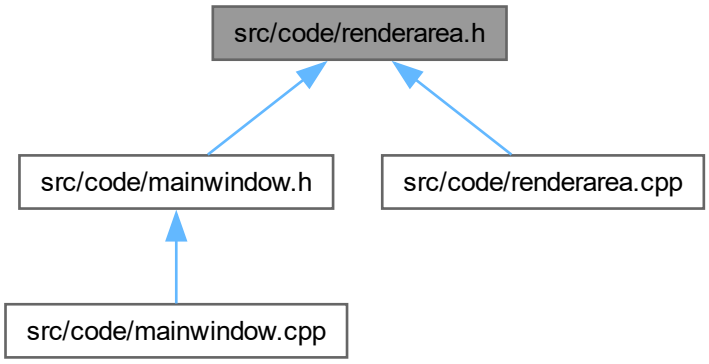
```
#include <QWidget>
#include <QPainter>
#include <QPen>
```

```
#include "../objects/all_shapes.h"
#include "../objects/vector.h"
#include <QMouseEvent>
```

Include dependency graph for renderarea.h:



This graph shows which files directly or indirectly include this file:



Classes

- class RenderArea

9.43 renderarea.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include <QWidget>
00004 #include <QPainter>
00005 #include <QPen>
00006
00007 #include "../objects/all_shapes.h"
00008 #include "../objects/vector.h"
00009 #include <QMouseEvent>
00010
00011 class RenderArea : public QWidget
00012 {
00013     Q_OBJECT
00014
00015 public:
```

```
00016     RenderArea(QWidget *parent = nullptr);
00017     void setRenderShapes(const alpha::vector<Shape*>* renderShapes);
00018     const alpha::vector<Shape*>& getShapes() const;
00019
00020     void mousePressEvent(QMouseEvent* event) override;
00021     void mouseMoveEvent(QMouseEvent* event) override;
00022     void mouseDoubleClickEvent(QMouseEvent* event) override;
00023     void mouseReleaseEvent(QMouseEvent* event) override;
00024
00025     void setShapeSelectedIndex(int newIndex);
00026     void resetSelection();
00027     void setEditPrivileges(bool edit);
00028
00029     void updateShapeDisplayCoords(Shape* item, const QPoint& position) const;
00030     int  getShapeSelected() const;
00031     int  getShapeSelectedIndex() const;
00032
00033 protected:
00034     void paintEvent(QPaintEvent *event) override;
00035
00036 private:
00037     const alpha::vector<Shape*>* renderShapes;      // Holds currently renderedShapes
00038     int shapeSelectedIndex; // This is the vector index of the current shape selected, this is done so
    we prevent multiple shapes being selected at once
00039     bool allowEditing = false; // by default you cannot edit shapes
00040 };
```

9.44 src/code/RenderAreaManager.cpp File Reference

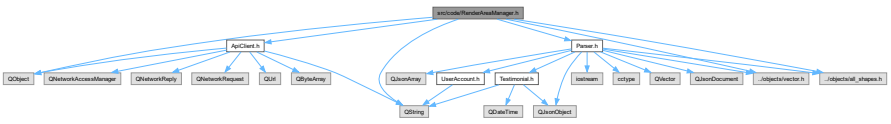
```
#include "RenderAreaManager.h"
#include <QString>
#include <QDebug>
#include <QBrush>
#include <QPen>
```

Include dependency graph for RenderAreaManager.cpp:

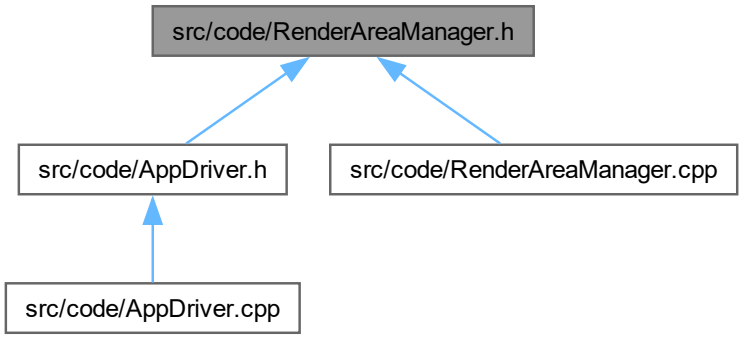
9.45 src/code/RenderAreaManager.h File Reference

```
#include <QObject>
#include "ApiClient.h"
#include "Parser.h"
#include <QString>
#include "../objects/all_shapes.h"
#include "../objects/vector.h"
```

Include dependency graph for RenderAreaManager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class RenderAreaManager

9.46 RenderAreaManager.h

Go to the documentation of this file.

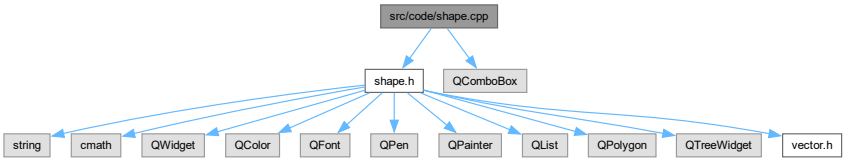
```
00001 #ifndef RENDER_AREA_MANAGER
00002 #define RENDER_AREA_MANAGER
00003 #include <QObject>
00004 #include "ApiClient.h"
00005 #include "Parser.h"
00006 #include <QString>
00007 #include "../objects/all_shapes.h"
00008 #include "../objects/vector.h"
00009
00010 class RenderAreaManager : public QObject {
00011     Q_OBJECT
00012 public:
00021     explicit RenderAreaManager(QObject* parent = nullptr);
00022     ~RenderAreaManager();
00023
00027     alpha::vector<Shape*>* getShapesRef();
00028
00037     void addShape(Shape* shape);
00038     void modifyShape(Shape* shape, QString key, int value);
00039     void modifyDisplayedText(Text* obj, QString newText);
00040     void deleteShape(const int trackerId);
00041     void deleteAllShapes();
00042     void loadShapes();
00043     void saveShapes();
00044
00045
00046 signals:
00055     void renderAreaChanged();
00056     void renderAreaNotChanged(const QString &message);
00057     void statusMessage(const QString &message);
00058
00059
00060 private slots:
00066     void onGoodGetResponse(const QString &json);
00067     void onBadGetResponse(const QString &errorMsg);
00068     void onGoodPostResponse();
00069     void onBadPostResponse(const QString &errorMsg);
00070     void onGoodDeleteResponse();
00071     void onBadDeleteResponse(const QString &errorMsg);
00072
```

```
00073
00074 private:
00075     alpha::vector<Shape*> renderedShapes;
00076     ApiClient client;
00077     Parser parse;
00078 };
00079
00080 #endif // RENDER_AREA_MANAGER
```

9.47 src/code/shape.cpp File Reference

```
#include "shape.h"
#include <QComboBox>
```

Include dependency graph for shape.cpp:



Functions

- bool operator== (const Shape &shape1, const Shape &shape2)
- bool operator< (const Shape &shape1, const Shape &shape2)

9.47.1 Function Documentation

9.47.1.1 operator<()

```
bool operator< (
            const Shape & shape1,
            const Shape & shape2)
```

Parameters

shape1	
shape2	

Returns

9.47.1.2 operator==()

```
bool operator== (
            const Shape & shape1,
            const Shape & shape2)
```

Parameters

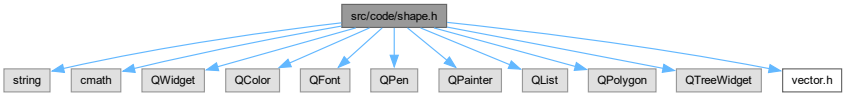
shape1	
shape2	

Returns

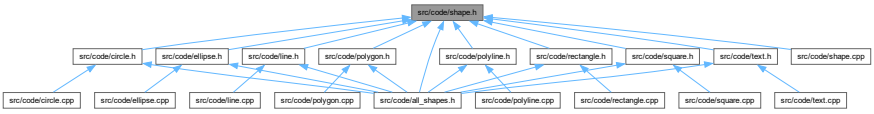
9.48 src/code/shape.h File Reference

```
#include <string>
#include <cmath>
#include <QWidget>
#include <QColor>
#include <QFont>
#include <QPen>
#include <QPainter>
#include <QList>
#include <QPolygon>
#include <QTreeWidget>
#include "vector.h"
```

Include dependency graph for shape.h:



This graph shows which files directly or indirectly include this file:



Classes

- interface Shape

  The Shape Abstract Base Class.

Variables

- const double PI = 3.14

9.48.1 Variable Documentation

9.48.1.1 PI

```
const double PI = 3.14
```

9.49 shape.h

Go to the documentation of this file.

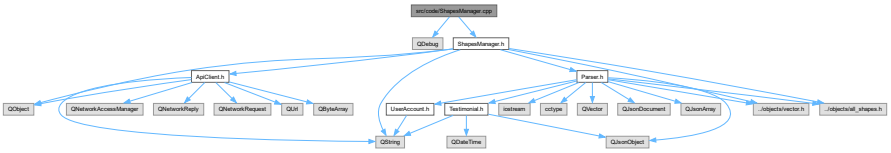
```
00001 #ifndef SHAPE_H
00002 #define SHAPE_H
00003
00004 #include <string>
00005 #include <cmath>
00006 #include <QWidget>
00007 #include <QColor>
00008 #include <QFont>
00009 #include <QPen>
00010 #include <QPainter>
00011 #include <QList>
00012 #include <QPolygon>
00013 #include <QTreeWidget>
00014 #include "vector.h"
00015
00016
00017 using std::string;
00018
00019 using Qt::GlobalColor;
00020 using Qt::PenCapStyle;
00021 using Qt::PenStyle;
00022 using Qt::PenJoinStyle;
00023 using Qt::BrushStyle;
00024 using Qt::AlignmentFlag;
00025
00026 const double PI = 3.14;
00027
00034
00035 class Shape
00036 {
00043     friend bool operator==(const Shape& shape1, const Shape& shape2);
00050     friend bool operator<(const Shape& shape1, const Shape& shape2);
00051
00052 public:
00053
00061     Shape(string shapeType,
00062           QPoint coords,
00063           QPen   pen,
00064           QBrush brush);
00065
00069     virtual ~Shape();
00070
00071     static int nextTracker[9];
00072     static bool trackersInUse[9000];
00073
00080     virtual void Draw(QWidget* renderArea) = 0;
00081
00087     virtual void Move(int x, int y);
00088
00095     virtual double Perimeter() const = 0;
00096
00103     virtual double Area()      const = 0;
00104
00112     virtual bool isPointInside(const QPoint& point) const = 0;
00113
00117     void CreateParentItem();
00118
00122     void CreatePenChild();
00123
00127     void CreateBrushChild();
00128
00133     void CreatePointsChild(const int POINTS_NUM);
00134
00136     int    getShapeId()   const;
00137     int    getTrackerId() const;
00138     string getShapeType() const;
00139     bool   getSelected()  const;
00140
```

```
00141     int getX() const;
00142     int getY() const;
00143 
00144     QPainter& getPainter();
00145     QTreeWidgetItem* getParentItem();
00146     alpha::vector<QTreeWidgetItem*>& getChildItems();
00147     alpha::vector<QTreeWidgetItem*>& getPointsItems();
00148     alpha::vector<QTreeWidgetItem*>& getPenItems();
00149     alpha::vector<QTreeWidgetItem*>& getBrushItems();
00150 
00151     int          getPenWidth()      const;
00152     PenStyle     getPenStyle()      const;
00153     PenCapStyle  getPenCapStyle()   const;
00154     PenJoinStyle getPenJoinStyle()  const;
00155     QColor       getPenColor()      const;
00156     QColor       getBrushColor()    const;
00157     BrushStyle   getBrushStyle()    const;
00158     QPen         getPen()           const;
00159     QBrush       getBrush()         const;
00160     QPoint       getPoints()        const;
00161     int          getChildEnd()      const;
00162     int          getPenItemsEnd()   const;
00163     int          getBrushItemsEnd() const;
00165 
00167     void setShapeId(int shapeId);
00168     void setShapeType(string shapeType);
00169     void setSelected(bool selected);
00170 
00171     void setTrackerId(int trackerId);
00172     void allocateTrackerId(int shapeId);
00173 
00174     void setX(int x);
00175     void setY(int y);
00176 
00177     void setPen(GlobalColor penColor, int penWidth, PenStyle penStyle, PenCapStyle penCapStyle,
      PenJoinStyle penJoinStyle);
00178     void setBrush(GlobalColor brushColor, BrushStyle brushStyle);
00179 
00180     // These functions make it easier to change pen and brush properties in
      RenderAreaManager::modifyShape()
00181     QPen& setInternalPen();
00182     QBrush& setInternalBrush();
00184 
00185 protected:
00186     QTreeWidgetItem* parentItem;
00187     alpha::vector<QTreeWidgetItem*> childItems;
00188     alpha::vector<QTreeWidgetItem*> pointsItems;
00189     alpha::vector<QTreeWidgetItem*> penItems;
00190     alpha::vector<QTreeWidgetItem*> brushItems;
00191 
00192 private:
00193     int      shapeId;
00194     int      trackerId;
00195     string   shapeType;
00196 
00197     QPen     pen;
00198     QBrush   brush;
00199     QPoint   coords;
00200 
00201     QPainter painter;
00202 
00203     bool isSelected = false;
00204 
00205     // Disable Copy Operations
00206     Shape(Shape& shape) = delete;
00207     Shape& operator=(Shape& object) = delete;
00208 };
00209 
00210 #endif // SHAPE_H
00211 
```

9.50 src/code/ShapesManager.cpp File Reference

```
#include <QDebug>
#include "ShapesManager.h"
```

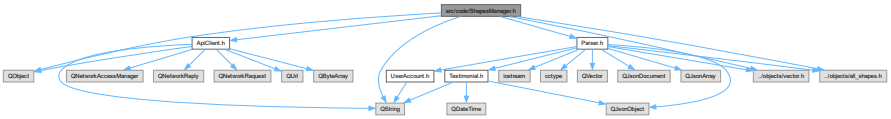
Include dependency graph for ShapesManager.cpp:



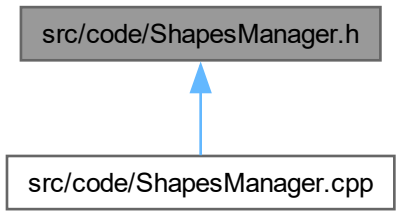
9.51 src/code/ShapesManager.h File Reference

```
#include <QObject>
#include <QString>
#include "ApiClient.h"
#include "Parser.h"
#include "../objects/all_shapes.h"
#include "../objects/vector.h"
```

Include dependency graph for ShapesManager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ShapesManager

9.52 ShapesManager.h

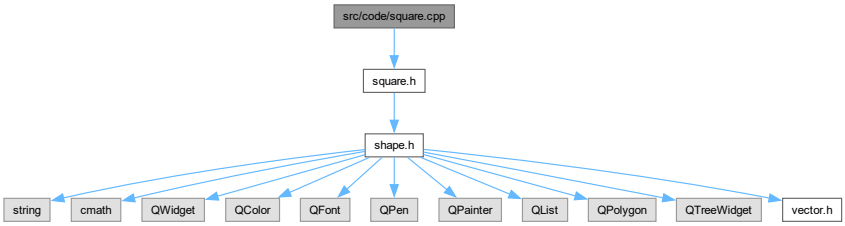
Go to the documentation of this file.

```
00001 #ifndef SHAPES_MANAGER
00002 #define SHAPES_MANAGER
00003
00004 #include <QObject>
00005 #include <QString>
00006 #include "ApiClient.h"
00007 #include "Parser.h"
00008 #include "../objects/all_shapes.h"
00009 #include "../objects/vector.h"
00010
00011 class ShapesManager : public QObject {
00012     Q_OBJECT
00013 public:
00022     explicit ShapesManager(QObject* parent = nullptr);
00023     ~ShapesManager();
00024
00028     alpha::vector<Shape*>* getShapesRef();
00029
00038     void addShape(Shape* shape);
00039     void modifyShape(Shape* shape);
00040     void deleteShape(const int trackerId);
00041     void deleteAllShapes();
00042     void loadShapes();
00043     void saveShapes();
00044
00045
00046
00047 signals:
00056     void shapesChanged();
00057     void shapesNotChanged(const QString &messsage);
00058     void statusMessage(const QString &message);
00059
00060
00061 private slots:
00067     void onGoodGetResponse(const QString &json);
00068     void onBadGetResponse(const QString &errorMsg);
00069     void onGoodPostResponse();
00070     void onBadPostResponse(const QString &errorMsg);
00071     void onGoodDeleteResponse();
00072     void onBadDeleteResponse(const QString &errorMsg);
00073
00074
00075 private:
00076     alpha::vector<Shape*> shapes;
00077     ApiClient client;
00078     Parser parse;
00079 };
00080
00081 #endif // SHAPES_MANAGER
```

9.53 src/code/square.cpp File Reference

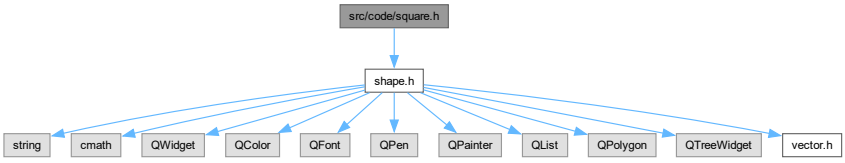
`#include "square.h"`

Include dependency graph for square.cpp:

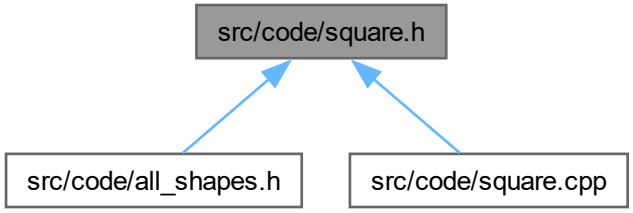


9.54 src/code/square.h File Reference

`#include "shape.h"`
Include dependency graph for square.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Square

9.55 square.h

Go to the documentation of this file.

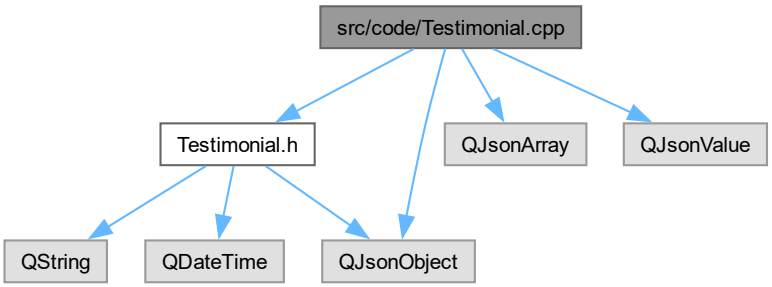
```
00001 #ifndef SQUARE_H
00002 #define SQUARE_H
00003
00004 #include "shape.h"
00005
00006 class Square : public Shape
00007 {
00008 public:
00009     Square(string shapeType,
00010            QPoint coords,
00011            QPen   pen,
00012            QBrush brush,
00013            int length);
00014
00015
00016     void Draw(QWidget* renderArea) override;
00017
00018     double Perimeter() const override;
00019     double Area()      const override;
00020
00021     bool isPointInside(const QPoint& point) const override;
```

```
00022
00023     int getLength() const;
00024
00025     void setLength(int newLength);
00026     void setX(int newX);
00027     void setY(int newY);
00028
00029 private:
00030     int length;
00031 };
00032
00033 #endif // SQUARE_H
```

9.56 src/code/Testimonial.cpp File Reference

```
#include "Testimonial.h"
#include <QJsonObject>
#include <QJsonArray>
#include <QJsonValue>
```

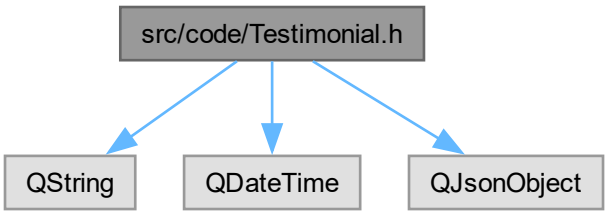
Include dependency graph for Testimonial.cpp:



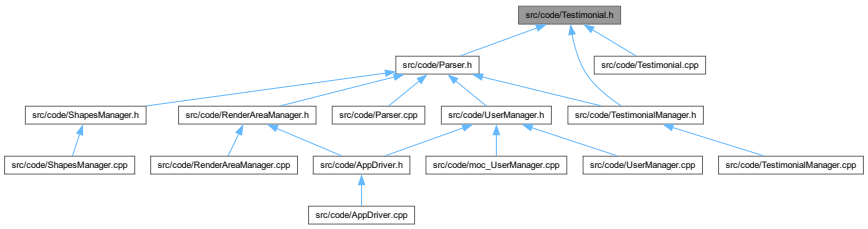
9.57 src/code/Testimonial.h File Reference

```
#include <QString>
#include <QDateTime>
#include <QJsonObject>
```

Include dependency graph for Testimonial.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Testimonial

9.58 Testimonial.h

Go to the documentation of this file.

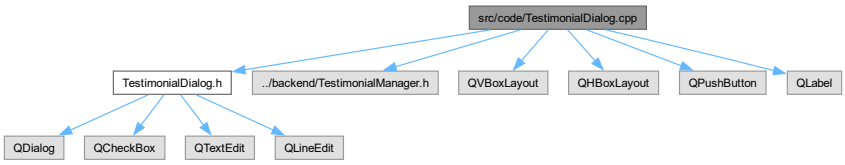
```
00001 #ifndef TESTIMONIAL_H
00002 #define TESTIMONIAL_H
00003
00004 #include <QString>
00005 #include <QDateTime>
00006 #include <QJsonObject>
00007
00008 class Testimonial {
00009 public:
00010     Testimonial(const QString& author = "", const QString& content = "", bool isGuest = true);
00011
00012     // getters for testimonial data
00013     QString getAuthor() const { return m_author; }
00014     QString getContent() const { return m_content; }
00015     QDateTime getTimestamp() const { return m_timestamp; }
00016     bool isGuest() const { return m_isGuest; }
00017     bool isSatisfactory() const { return m_isSatisfactory; }
00018
00019     // setter for satisfaction flag
00020     void setIsSatisfactory(bool value) { m_isSatisfactory = value; }
00021
00022     // json conversion for database storage
00023     QJsonObject toJson() const;
00024     static Testimonial fromJson(const QJsonObject& json);
00025
00026 private:
00027     QString m_author;        // name of person giving testimonial
00028     QString m_content;       // actual testimonial text
00029     QDateTime m_timestamp;   // when it was submitted
00030     bool m_isGuest;         // if they're a guest or registered
00031     bool m_isSatisfactory;   // if we should show this one
00032 };
00033
00034 #endif // TESTIMONIAL_H
```

9.59 src/code/TestimonialDialog.cpp File Reference

```
#include "TestimonialDialog.h"
#include "../backend/TestimonialManager.h"
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <QPushButton>
```

```
#include <QLabel>
```

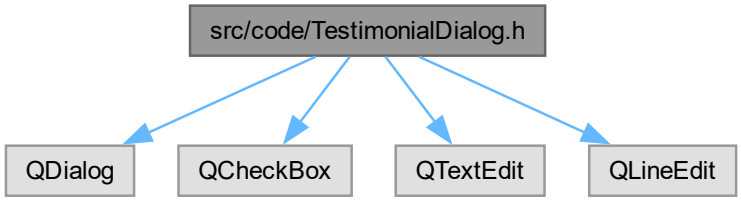
Include dependency graph for TestimonialDialog.cpp:



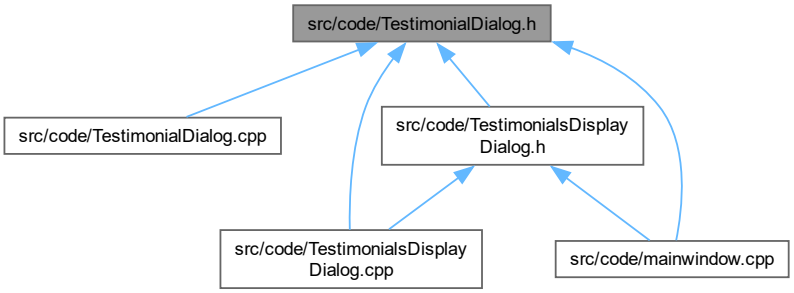
9.60 src/code/TestimonialDialog.h File Reference

```
#include <QDialog>
#include <QCheckBox>
#include <QTextEdit>
#include <QLineEdit>
```

Include dependency graph for TestimonialDialog.h:



This graph shows which files directly or indirectly include this file:



Classes

- class TestimonialDialog

9.61 TestimonialDialog.h

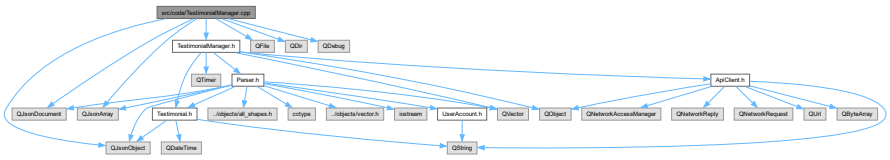
Go to the documentation of this file.

```
00001 #ifndef TESTIMONIALDIALOG_H
00002 #define TESTIMONIALDIALOG_H
00003
00004 #include <QDialog>
00005 #include <QCheckBox>
00006 #include <QTextEdit>
00007 #include <QLineEdit>
00008
00009 // dialog for submitting new testimonials
00010 class TestimonialDialog : public QDialog {
00011     Q_OBJECT
00012
00013 public:
00014     explicit TestimonialDialog(QWidget* parent = nullptr);
00015
00016 private slots:
00017     void onSubmit();
00018     void onCancel();
00019
00020 private:
00021     QLineEdit* m_authorEdit;     // name input
00022     QTextEdit* m_contentEdit;    // testimonial input
00023     QCheckBox* m_doNotShowAgain; // checkbox for future prompts
00024 };
00025
00026 #endif // TESTIMONIALDIALOG_H
```

9.62 src/code/TestimonialManager.cpp File Reference

```
#include "TestimonialManager.h"
#include <QFile>
#include <QJsonDocument>
#include <QJsonArray>
#include <QJsonObject>
#include <QDir>
#include <QDebug>
```

Include dependency graph for TestimonialManager.cpp:

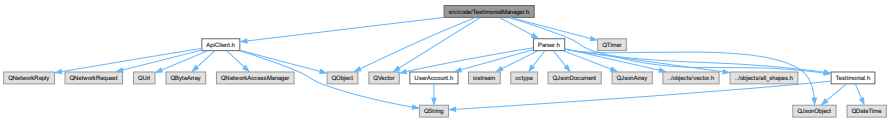


9.63 src/code/TestimonialManager.h File Reference

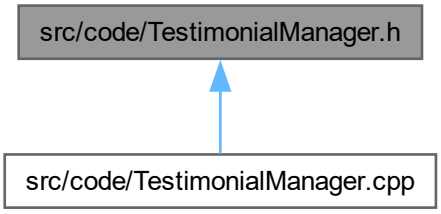
```
#include "Testimonial.h"
#include <QObject>
#include <QVector>
#include <QTimer>
#include "ApiClient.h"
```

```
#include "Parser.h"
```

Include dependency graph for TestimonialManager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class TestimonialManager

9.64 TestimonialManager.h

Go to the documentation of this file.

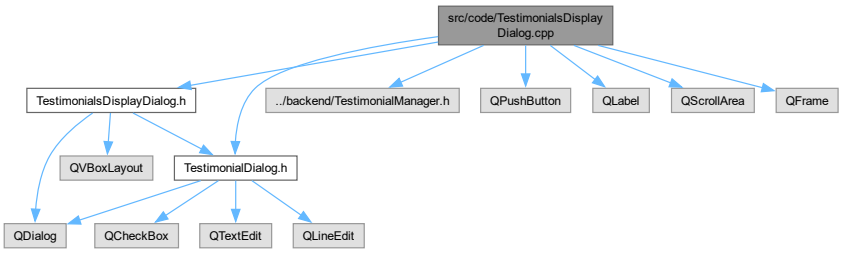
```
00001 #ifndef TESTIMONIALMANAGER_H
00002 #define TESTIMONIALMANAGER_H
00003
00004 #include "Testimonial.h"
00005 #include <QObject>
00006 #include <QVector>
00007 #include <QTimer>
00008 #include "ApiClient.h"
00009 #include "Parser.h"
00010
00011 class TestimonialManager : public QObject {
00012     Q_OBJECT
00013 public:
00014     static TestimonialManager& getInstance();
00015
00016     // testimonial management functions
00017     void addTestimonial(const Testimonial& testimonial);
00018     QVector<Testimonial> getSatisfactoryTestimonials() const;
00019     bool hasUserGivenTestimonial(const QString& username) const;
00020
00021     // handle do not show again preference
00022     void setDoNotShowAgain(const QString& username, bool value);
00023     bool getDoNotShowAgain(const QString& username) const;
00024
00025     // time tracking functions
00026     void startTrackingTime();
00027     void stopTrackingTime();
00028
00029 signals:
```

```
00030     void shouldPromptForTestimonial();
00031
00032 private slots:
00033     // Callbacks for API client responses
00034     void onGoodGetResponse(const QString& json);
00035     void onBadGetResponse(const QString& error);
00036     void onGoodPostResponse();
00037     void onBadPostResponse(const QString& error);
00038
00039 private:
00040     TestimonialManager();
00041     ~TestimonialManager();
00042
00043     // database operations
00044     void loadTestimonials();
00045     void saveTestimonials();
00046     void checkTimeAndPrompt();
00047
00048     QVector<Testimonial> m_testimonials;          // stored testimonials
00049     QTimer* m_trackingTimer;                     // tracks user time
00050     QHash<QString, bool> m_doNotShowAgain;       // stores user preferences
00051     QHash<QString, int> m_userTimeTracking;      // tracks time per user
00052
00053     ApiClient client;
00054     Parser parse;
00055
00056     // timing constants
00057     static const int INITIAL_PROMPT_TIME = 30 * 60;    // 30 min in seconds
00058     static const int REPEAT_PROMPT_TIME = 60 * 60;     // 1 hour in seconds
00059 };
00060
00061 #endif // TESTIMONIALMANAGER_H
```

9.65 src/code/TestimonialsDisplayDialog.cpp File Reference

```
#include "TestimonialsDisplayDialog.h"
#include "../backend/TestimonialManager.h"
#include "TestimonialDialog.h"
#include <QPushButton>
#include <QLabel>
#include <QScrollArea>
#include <QFrame>
```

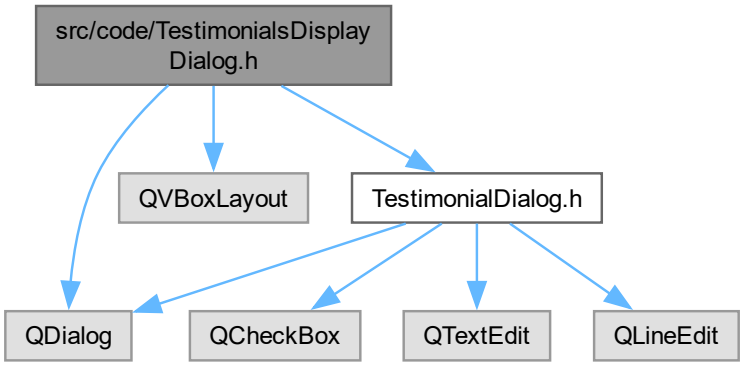
Include dependency graph for TestimonialsDisplayDialog.cpp:



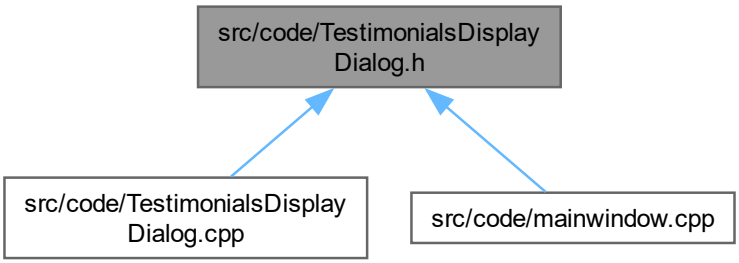
9.66 src/code/TestimonialsDisplayDialog.h File Reference

```
#include <QDialog>
#include <QVBoxLayout>
```

`#include "TestimonialDialog.h"`
Include dependency graph for TestimonialsDisplayDialog.h:



This graph shows which files directly or indirectly include this file:



Classes

- class TestimonialsDisplayDialog

9.67 TestimonialsDisplayDialog.h

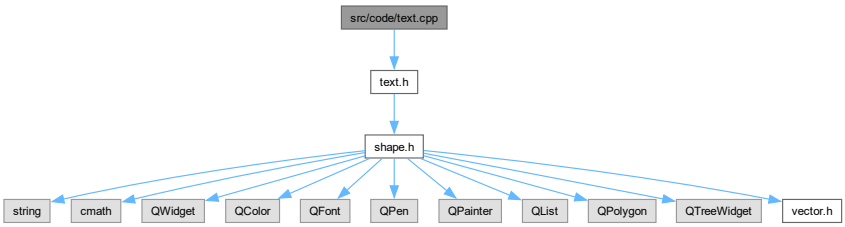
Go to the documentation of this file.

```
00001 #ifndef TESTIMONIALSDISPLAYDIALOG_H
00002 #define TESTIMONIALSDISPLAYDIALOG_H
00003
00004 #include <QDialog>
00005 #include <QVBoxLayout>
00006 #include "TestimonialDialog.h"
00007
00008 // dialog to show all testimonials
00009 class TestimonialsDisplayDialog : public QDialog {
```

```
00010     Q_OBJECT
00011
00012 public:
00013     explicit TestimonialsDisplayDialog(QWidget* parent = nullptr);
00014
00015 private:
00016     void refreshTestimonials();
00017     QVBoxLayout* m_testimonialsLayout;
00018 };
00019
00020 #endif // TESTIMONIALSDISPLAYDIALOG_H
```

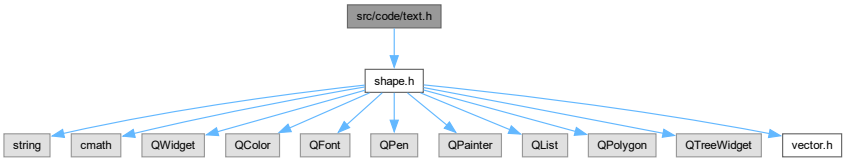
9.68 src/code/text.cpp File Reference

`#include "text.h"`
Include dependency graph for text.cpp:

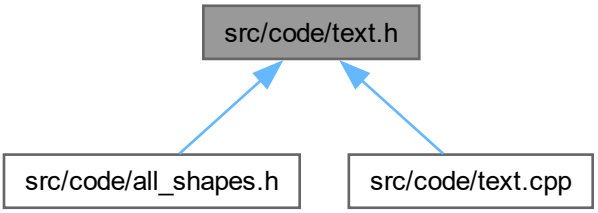


9.69 src/code/text.h File Reference

`#include "shape.h"`
Include dependency graph for text.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Text

9.70 text.h

Go to the documentation of this file.

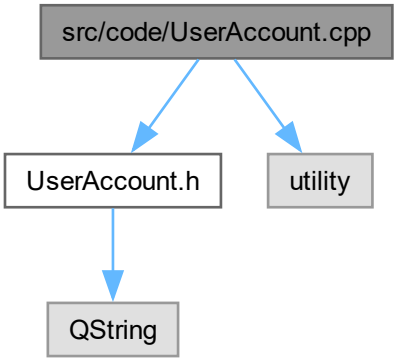
```
00001 #ifndef TEXT_H
00002 #define TEXT_H
00003
00004 #include "shape.h"
00005
00006 class Text : public Shape
00007 {
00008 public:
00009     Text(string shapeType,
00010          QPoint coords,
00011          QString textString,
00012          GlobalColor   textColor,
00013          AlignmentFlag textAlignment,
00014          QFont font,
00015          int length,
00016          int width);
00017
00018
00019     void Draw(QWidget* renderArea) override;
00020
00021     double Perimeter() const override;
00022     double Area()      const override;
00023
00024     bool isPointInside(const QPoint& point) const override;
00025
00026     void setText(QString text);
00027     void setLength(int newLength);
00028     void setWidth(int newWidth);
00029     void setX(int newX);
00030     void setY(int newY);
00031     void setAlignment(Qt::AlignmentFlag alignment);
00032
00033     QFont& setInternalFont();
00034
00035     /************* ACCESSOR FUNCTIONS *************/
00036     int           getLength()        const;
00037     int           getWidth()         const;
00038     QString       getTextString()    const;
00039     GlobalColor   getTextColor()     const;
00040     QFont         getFont()          const;
00041     AlignmentFlag getTextAlignment() const;
00042     int           getFontStyle()     const;
00043     QFont::Weight getFontWeight()    const;
00044     /*********************************************/
00045
```

```
00046 private:
00047     int length;
00048     int width;
00049
00050     QString       textString;
00051     GlobalColor   textColor;
00052     QFont         font;
00053     AlignmentFlag textAlignment;
00054 };
00055
00056 #endif // TEXT_H
```

9.71 src/code/UserAccount.cpp File Reference

```
#include "UserAccount.h"
#include <utility>
```

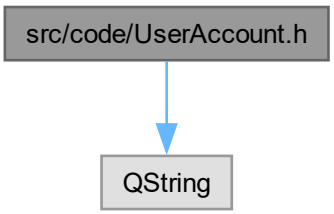
Include dependency graph for UserAccount.cpp:



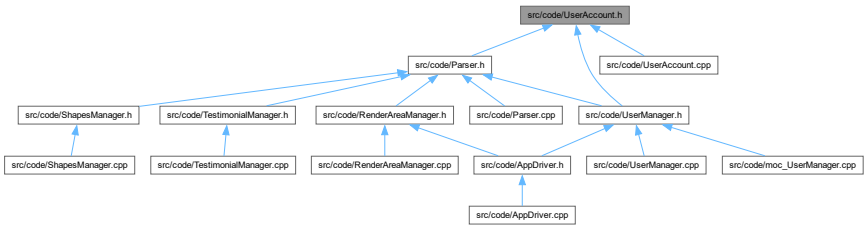
9.72 src/code/UserAccount.h File Reference

```
#include <QString>
```

Include dependency graph for UserAccount.h:



This graph shows which files directly or indirectly include this file:



Classes

- class UserAccount

9.73 UserAccount.h

Go to the documentation of this file.

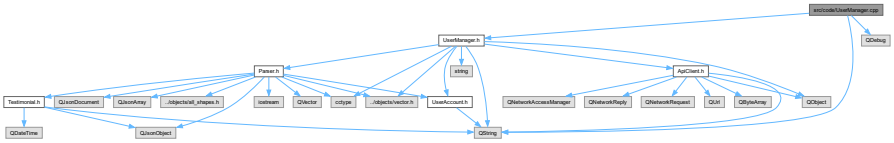
```
00001 #ifndef USER_ACCOUNT_H
00002 #define USER_ACCOUNT_H
00003
00004 #include <QString>
00005
00006 class UserAccount {
00007 public:
00008     // Constructors
00009     UserAccount();      //creates a default guest account
00010     UserAccount(QString username, QString password, bool admin);
00011     ~UserAccount();
00012
00013     // Copy and move constructors and assignment operators
00014     UserAccount(const UserAccount& other);
00015     UserAccount& operator=(const UserAccount& other);
00016     UserAccount(UserAccount&& other) noexcept;
00017     UserAccount& operator=(UserAccount&& other) noexcept;
00018
00019     //Getters
00020     QString getUsername() const;
00021     QString getPassword() const;
00022     bool isAdmin() const;
00023
00024     //Setters
00025     void setUsername(QString& username);
00026     void setPassword(QString& password);
00027     void setAdmin(bool admin);
00028     void setUserAccount(QString& username, QString& password, bool admin);
00029
00030
00031 private:
00032     QString username;
00033     QString password;
00034     bool admin;
00035 };
00036
00037 #endif // USER_ACCOUNT_H
```

9.74 src/code/UserManager.cpp File Reference

```
#include "UserManager.h"
#include <QString>
```

```
#include <QDebug>
```

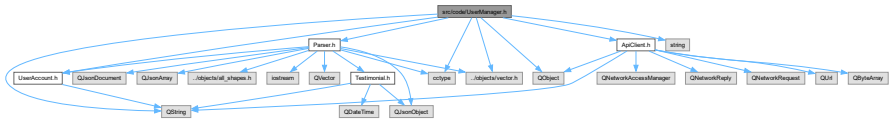
Include dependency graph for UserManager.cpp:



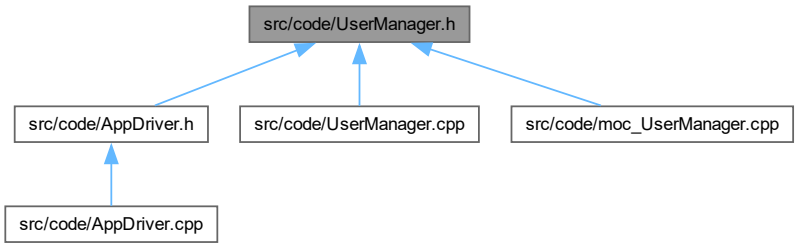
9.75 src/code/UserManager.h File Reference

```
#include <QObject>
#include <QString>
#include <cctype>
#include <string>
#include "ApiClient.h"
#include "Parser.h"
#include "UserAccount.h"
#include "../objects/vector.h"
```

Include dependency graph for UserManager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class UserManager

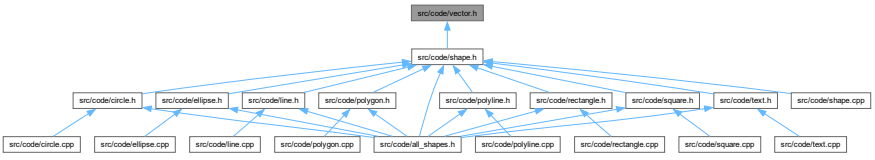
9.76 UserManager.h

Go to the documentation of this file.

```
00001 #ifndef USER_MANAGER_H
00002 #define USER_MANAGER_H
00003 #include <QObject>
00004 #include <QString>
00005 #include <cctype>
00006 #include <string>
00007 #include "ApiClient.h"
00008 #include "Parser.h"
00009 #include "UserAccount.h"
00010 #include "../objects/vector.h"
00011
00012 class UserManager : public QObject {
00013     Q_OBJECT
00014 public:
00023     explicit UserManager(QObject* parent = nullptr);
00024     ~UserManager();
00025
00029     UserAccount* getCurrUserRef();
00030
00039     void addUser(const QString username, const QString password, const bool admin);
00040     void modifyUser(const QString username, const QString password, const bool admin);
00041     void deleteUser(QString username);
00042     void deleteAllUsers();
00043     void loadUsers();
00044     void saveUsers();
00045     void authenticate(const QString username, const QString password);
00046
00047
00048 signals:
00055     void userChanged();
00056     void userNotChanged(const QString &message);
00057     void statusMessage(const QString &message);
00058     void userAuthenticated(const UserAccount* currUser);
00059     void authenticationFailed(const QString &message);
00060
00061
00062 private slots:
00063
00064
00069     void onGoodGetResponse(const QString &json);
00070     void onBadGetResponse(const QString &errorMsg);
00071     void onGoodPostResponse();
00072     void onBadPostResponse(const QString &errorMsg);
00073     void onGoodDeleteResponse();
00074     void onBadDeleteResponse(const QString &errorMsg);
00075
00076
00077 private:
00078     //Data Members
00079     UserAccount* currUser;
00080     alpha::vector<UserAccount*> users;
00081     ApiClient client;
00082     Parser parse;
00083 };
00084
00085 #endif // USER_MANAGER_H
```

9.77 src/code/vector.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class alpha::vector< T >

Namespaces

- namespace alpha

Macros

- #define ALPHA_VECTOR_H

9.77.1 Macro Definition Documentation

9.77.1.1 ALPHA_VECTOR_H

```
#define ALPHA_VECTOR_H
```

9.78 vector.h

Go to the documentation of this file.

```
00001 #ifndef ALPHA_VECTOR_H
00002 #define ALPHA_VECTOR_H  //This is necessary for inclusion guards. DO NOT DELETE
00003 /*****************************************************************************
00004  * vector.h
00005  * ---------------------------------------------------------------------------
00006  * Worked on by: Aram, Aspen, Luke
00007  * ---------------------------------------------------------------------------
00008  * This vector will be used to hold the shapes that will be displayed.
00009  * It supports the following basic operations:
00010  *      - constructors for one or more arguments
00011  *      - default constructors
00012  *      - copy constructor
00013  *      - copy assignment
00014  *      - move constructor
00015  *      - move assignment
00016  *      - destructor
00017  *
00018  * Vector also supports:
00019  *      - a basic iterator member type and member function
00020  *      - begin()
00021  *      - end() operations
00022 *****************************************************************************/
00023 namespace alpha {
00024 template<class T>
00025
00026 class vector
00027 {
00028     int size_v;     // the size
00029     T* elem;        // a pointer to the elements
00030     int space;      // size + free_space
00031
00032     public:
00033         /*************************************************************************
00034          * DEFAULT CONSTRUCTOR
00035          *************************************************************************/
00036         vector() : size_v(0), space(1) {
00037             elem = new T[space];
00038         } // END vector()
00039
00040         /*************************************************************************
00041          * ALTERNATE CONSTRUCTOR - size given
00042          *************************************************************************/
00043         explicit vector(int s) : size_v(s), space(s) {
00044             elem = new T[space];
```

```
00045         } // END explicit vector(int s)
00046
00047         /*************************************************************************
00048          * COPY CONSTRUCTOR
00049          *************************************************************************/
00050         vector(const vector& other) : size_v(other.size_v), space(other.space) {
00051             elem = new T[space];
00052
00053             for (int i = 0; i < size_v; i++) {
00054                 elem[i] = other.elem[i];
00055             } // END for (int i = 0; i < size_v; i++)
00056         } // END vector(const vector& other)
00057
00058         /*************************************************************************
00059          * COPY ASSIGNMENT
00060          *************************************************************************/
00061         vector& operator=(const vector& other) {
00062             /*********************************************************************
00063             * CHECKS IF SELF-ASSIGNMENT
00064             *********************************************************************/
00065             if (this == &other) {
00066                 return *this;
00067             }
00068
00069             /*********************************************************************
00070             * IF NOT SELF-ASSIGNMENT
00071             *********************************************************************/
00072             delete[] elem;
00073
00074             size_v = other.size_v;
00075             space = other.space;
00076             elem = new T[space];
00077
00078             for (int i = 0; i < size_v; i++) {
00079                 elem[i] = other.elem[i];
00080             }
00081
00082             return *this;
00083         } // END vector& operator=vector
00084
00085         /*************************************************************************
00086          * MOVE CONSTRUCTOR
00087          *************************************************************************/
00088         vector(vector&& other) noexcept
00089             : size_v(other.size_v), elem(other.elem), space(other.space) {
00090             other.size_v = 0;
00091             other.elem = nullptr;
00092             other.space = 0;
00093         }
00094
00095         /*************************************************************************
00096          * MOVE ASSIGNMENT
00097          *************************************************************************/
00098         vector& operator=(vector&& other) noexcept {
00099             /*********************************************************************
00100             * CHECKS IF SELF-ASSIGNMENT
00101             *********************************************************************/
00102             if (this == &other) {
00103                 return *this;
00104             }
00105
00106             /*********************************************************************
00107             * IF NOT SELF-ASSIGNMENT
00108             *********************************************************************/
00109             delete[] elem;
00110
00111             size_v = other.size_v;
00112             space = other.space;
00113             elem = other.elem;
00114
00115             other.size_v = 0;
00116             other.space = 0;
00117             other.elem = nullptr;
00118
00119            return *this;
00120         } // END vector& operator=(const vector&& other) noexcept
00121
00122         /*************************************************************************
00123          * DESTRUCTOR
00124          *************************************************************************/
00125         ~vector() { delete[] elem; }
00126
00127         /*************************************************************************
00128          * ACCESSOR - RETURN REFERENCE - MODIFIABLE
00129          *************************************************************************/
00130         T& operator[] (int n) { return elem[n]; }
00131
```

```
00132         /*************************************************************************
00133          * ACCESSOR - RETURN REFERENCE
00134          *************************************************************************/
00135         const T& operator[] (int n) const { return elem[n]; }
00136
00137         /*************************************************************************
00138          * ACCESSOR - RETURN CURRENT SIZE
00139          *************************************************************************/
00140         int size() const { return size_v; }
00141
00142         /*************************************************************************
00143          * ACCESSOR - RETURN CURRENT AVAILABLE SPACE
00144          *************************************************************************/
00145         int capacity() const { return space; }
00146
00147         /*************************************************************************
00148          * void resize(int newsize)
00149          * -----------------------------------------------------------------------
00150          * This function will be passed a number:
00151          *      newsize     - size to increase vector by
00152          *
00153          * depending on size_v the following will happen:
00154          *      size_v = newsize
00155          *      nothing will happen
00156          *
00157          *      size_v < newsize
00158          *      will change size_v to newsize
00159          *
00160          *      size_v > newsize (default)
00161          *      error will occur
00162          * -----------------------------------------------------------------------
00163          * PRE-CONDITIONS
00164          *      size_v  - original size of vector
00165          *
00166          * POST-CONDITIONS
00167          *      newsize  - new size of vector
00168          *************************************************************************/
00169         void resize(int newsize) {
00170
00171             /*********************************************************************
00172             * DOES NOTHING - EQUALS EACH OTHER
00173             *********************************************************************/
00174             if (size_v == newsize) {}
00175
00176             /*********************************************************************
00177             * OG SIZE LESS THAN NEW SIZE
00178             *********************************************************************/
00179             else if (size_v < newsize) {
00180                 reserve(newsize);
00181                 size_v = newsize;
00182             } // END else if (size_v < newsize)
00183
00184             /*********************************************************************
00185             * DOES NOTHING - DEFAULT - SMALLER THAN OG / INVALID INPUT / ERROR
00186             *********************************************************************/
00187             else { }
00188
00189         } // END void resize(int newsize)
00190
00191         /*************************************************************************
00192          * FUNCTION - ADD ELEMENT
00193          * -----------------------------------------------------------------------
00194          * This function will be passed a value:
00195          *      val     - data to add to vector
00196          *
00197          * Function adds the data to the back of the vector. If there is no
00198          * space then the function will increase the space before adding
00199          * -----------------------------------------------------------------------
00200          * PRE-CONDITIONS
00201          *      size_v  - original size of vector
00202          *
00203          * POST-CONDITIONS
00204          *      newsize  - new size of vector
00205          *************************************************************************/
00206         void push_back(const T val) {
00207             /*********************************************************************
00208              * IF NO SPACE - MAKE ROOM
00209              *********************************************************************/
00210             if (size_v == space) {
00211                 reserve(space + 1);
00212             }
00213             /*********************************************************************
00214              * IF SPACE
00215              *********************************************************************/
00216             elem[size_v] = val;
00217             size_v++;
00218         } // END void push_back(const T val)
```

```
00219
00220          /*************************************************************************
00221           * void reserve(int newalloc)
00222           * -----------------------------------------------------------------------
00223           * This function will be passed a number:
00224           *      newalloc     - size to increase vector capacity by
00225           *
00226           * depending on space the following will happen:
00227           *      space = newalloc
00228           *      nothing will happen
00229           *
00230           *      space < newalloc
00231           *      will change space to newalloc
00232           *
00233           *      size_v > newalloc (default)
00234           *      error will occur
00235           * -----------------------------------------------------------------------
00236           * PRE-CONDITIONS
00237           *      space  - original size of vector
00238           *
00239           * POST-CONDITIONS
00240           *      newalloc  - new size of vector
00241           *************************************************************************/
00242          void reserve(int newalloc) {
00243              /*********************************************************************
00244              * DOES NOTHING - EQUALS EACH OTHER
00245              *********************************************************************/
00246              if (space == newalloc) {}
00247
00248              /*********************************************************************
00249              * OG SIZE LESS THAN NEW SIZE
00250              *********************************************************************/
00251              else if (space < newalloc) {
00252
00253                  // CREATE BIGGER ARRAY
00254                  T* new_elem = new T[newalloc];
00255
00256                  // COPY DATA
00257                  for (int i = 0; i < size_v; i++) {
00258                      new_elem[i] = elem[i];
00259                  }
00260
00261                  delete[] elem;
00262                  elem = new_elem;
00263
00264                  // UPDATE SPACE
00265                  space = newalloc;
00266              } // END else if (space < newalloc)
00267
00268              /*********************************************************************
00269              * DOES NOTHING - DEFAULT - SMALLER THAN OG / INVALID INPUT / ERROR
00270              *********************************************************************/
00271              else { }
00272
00273          } // END void reserve(int newalloc)
00274
00275
00276          using iterator = T*;
00277          using const_iterator = const T*;
00278
00279          /*************************************************************************
00280           * POINTS TO FIRST ELEMENT
00281           *************************************************************************/
00282          iterator begin() { return elem; }
00283
00284          /*************************************************************************
00285           * CONSTANT - POINTS TO FIRST ELEMENT
00286           *************************************************************************/
00287          const_iterator begin() const { return elem; }
00288
00289          /*************************************************************************
00290           * POINTS TO ONE BEYOND THE LAST ELEMENT
00291           *************************************************************************/
00292          iterator end() { return elem + size_v; }
00293
00294          /*************************************************************************
00295           * CONSTANT - POINTS TO ONE BEYOND THE LAST ELEMENT
00296           *************************************************************************/
00297          const_iterator end() const { return elem + size_v; }
00298
00299          /*************************************************************************
00300           * INSERT NEW ELEMENT (V) BEFORE P
00301           *************************************************************************/
00302          iterator insert(iterator p, const T& v) {
00303              /*********************************************************************
00304               * CHECK IF ENOUGH SPACE
00305               *********************************************************************/
```

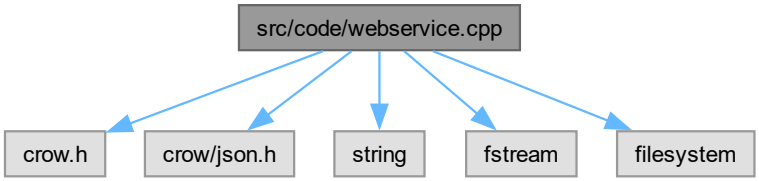
```
00306             // record the index where we want to insert
00307             int index = static_cast<int>(p - elem);
00308 
00309             // grow storage if needed
00310             if (size_v == space) {
00311                 reserve(space + 1);
00312             }
00313 
00314             // iterator next = end();
00315             // while (next != p) {
00316             //     *next = *(next - 1);
00317             //     next--;
00318             // }
00319 
00320             // recompute pointers into the (possibly moved) data
00321             p = elem + index;
00322             iterator end_it = elem + size_v;
00323 
00324             // shift elements [index..size_v-1] one position to the right
00325             for (iterator it = end_it; it != p; --it)
00326                 *it = *(it - 1);
00327 
00328             *p = v;
00329             size_v++;
00330 
00331             return p;
00332         } // END iterator insert(iterator p, const T& v)
00333 
00334         /**************************************************************************
00335          * REMOVE ELEMENT POINTED TO BY P
00336          **************************************************************************/
00337         iterator erase(iterator p) {
00338             /**************************************************************************
00339              * IF P AT THE END
00340              **************************************************************************/
00341             if (p == end()) {
00342                 return p;
00343             }
00344 
00345             iterator next = p + 1;
00346             while (next != end()) {
00347                 *p = *next;
00348                 p++;
00349                 next++;
00350             }
00351 
00352             p--;
00353             p->~T();
00354             size_v--;
00355 
00356             return p;
00357         } // END iterator erase(iterator p)
00358 
00359 }; // END class vector
00360 }; // END namespace alpha
00361 #endif // ALPHA_VECTOR_H
```

9.79 src/code/webservice.cpp File Reference

Implements the Crow web service for handling shapes and render area data.

```
#include <crow.h>
#include <crow/json.h>
#include <string>
#include <fstream>
#include <filesystem>
```

Include dependency graph for webservice.cpp:



Functions

- crow::json::rvalue get_json_file (const std::string &path)
  Reads and parses a JSON file.
- int main ()

9.79.1 Detailed Description

Implements the Crow web service for handling shapes and render area data.

This web service provides the following endpoints:

- TEST GET "/" : A test endpoint that returns a greeting.
- GET /shapes : Returns the JSON-formatted shapes data from ../../database/shapes.json.
- GET /render_area : Returns the JSON-formatted render area data from ../../database/render_area.json.
- GET /users : Returns the JSON-formatted user data from ../../database/users.json.
- POST /shapes : Accepts JSON data to update the shapes file (../../database/shapes.json).
- POST /render_area : Accepts JSON data to update the render area file (../../database/render_area.json).
- POST /users : Accepts JSON data to update the user file (../../database/users.json).
- DELETE /shapes-all : Clears all shapes data by truncating the shapes.json file.
- DELETE /render_area-all : Clears all render area data by truncating the render_area.json file.
- DELETE /users-all : Clears all user data by truncating the users.json file.

Note

Uses the Crow framework for handling HTTP requests. Make sure the database directory exists or is created.

9.79.2 Function Documentation

9.79.2.1 get_json_file()

```
crow::json::rvalue get_json_file (
            const std::string & path)
```

Reads and parses a JSON file.

Opens the file at the specified path, loads its contents into a stringstream, and uses Crow's JSON parser to convert the data into a crow::json::rvalue. Throws a std::runtime_error if the file cannot be opened or parsed.

Parameters

path	The file system path to the JSON file.

Returns

crow::json::rvalue The parsed JSON data.

Here is the caller graph for this function:

9.79.2.2 main()

```
int main ()
```

Test endpoint.

Returns a simple greeting message to verify that the web service is running.

Retrieves shapes data.

Reads the JSON data from "../../database/shapes.json", parses it using Crow's JSON parser, and returns the data with the Content-Type header set to "application/json".

Retrieves render area data.

Reads JSON data from "../../database/render_area.json", parses it using Crow's JSON parser, and returns the data with the Content-Type header set to "application/json".

Retrieves user account data.

Reads the JSON data from "../../database/user.json", parses it, and returns it with Content-Type "application/json".

Updates shapes.json with new shapes.

Accepts a POST request with JSON data in the body and writes the contents to the file "../../database/shapes.json". If the file cannot be opened, a 500 response is returned.

Updates the render area data.

Accepts a POST request with JSON data and writes it to the file "../../database/render_area.json". Returns a 500 response if the file cannot be opened.

Updates user.json with new user data.

Accepts a POST request with JSON body and writes it to "../../database/user.json".

Clears all shapes.

Deletes the shapes.json file content by truncating the file, leaving it empty (an empty JSON array).

Clears all render area data.

Deletes the render_area.json file content by truncating the file, leaving it empty (an empty JSON array).

Clears all user account data.

Deletes the user.json file content by truncating it, leaving an empty JSON array.

Retrieves testimonial data.

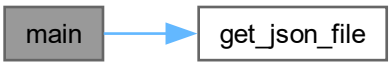
Reads the JSON array from "../../database/testimonials.json", sets Content-Type to "application/json", and returns the data. Returns 500 if file cannot be opened or parsed.

Creates or updates testimonial data.

Accepts a POST with JSON body containing an array of testimonials or a single testimonial object. Writes the body directly to "../../database/testimonials.json", creating the directory if needed. Returns 200 on success or 500 on failure.

Clears all testimonial data.

Truncates the testimonials.json file, writing an empty JSON array. Returns 200 on success or 500 on failure.Here is the call graph for this function:



Index

> Shapes, 11
~AppDriver
    AppDriver, 24
~MainWindow
    MainWindow, 62
~Parser
    Parser, 72
~RenderAreaManager
    RenderAreaManager, 114
~Shape
    Shape, 125
~ShapesManager
    ShapesManager, 152
~UserAccount
    UserAccount, 185
~UserManager
    UserManager, 191
~vector
    alpha::vector< T >, 198

addShape
    RenderAreaManager, 115
    ShapesManager, 152
addTestimonial
    TestimonialManager, 169
addUser
    UserManager, 192
all_shapes.h
    CIRCLE, 206
    ELLIPSE, 206
    LINE, 206
    POLYGON, 206
    POLYLINE, 206
    RECTANGLE, 206
    ShapeIDs, 205
    SQUARE, 206
    TEXT, 206
allocateTrackerId
    Shape, 126
alpha, 13
alpha::vector< T >, 196
    ~vector, 198
    begin, 198
    capacity, 199
    const_iterator, 197
    end, 199
    erase, 199
    insert, 199
    iterator, 197
    operator=, 200
    operator[], 200, 201
    push_back, 201
    reserve, 201
    resize, 202
    size, 202
    vector, 197, 198
ALPHA_VECTOR_H
    vector.h, 256
ApiClient, 15
    ApiClient, 17
    BadDeleteReply, 17
    BadGetReply, 18
    BadPostReply, 18
    DeleteRenderAreaAll, 19
    DeleteShapesAll, 19
    DeleteTestimonialsAll, 19
    DeleteUsersAll, 19
    GetRenderArea, 19
    GetShapes, 20
    GetTestimonials, 20
    GetUsers, 20
    GoodDeleteReply, 20
    GoodGetReply, 20
    GoodPostReply, 21
    PostRenderArea, 21
    PostShapes, 21
    PostTestimonials, 22
    PostUsers, 22
AppDriver, 22
    ~AppDriver, 24
    AppDriver, 24
    loadAllData, 24
    run, 24
    shutdown, 25
Area
    Circle, 30
    Ellipse, 41
    Line, 50
    Polygon, 81
    Polyline, 92
    Rectangle, 102
    Shape, 126
    Square, 160
    Text, 177
authenticate
    UserManager, 192
authenticationFailed
    UserManager, 192

BadDeleteReply

ApiClient, 17
BadGetReply
ApiClient, 18
BadPostReply
ApiClient, 18
begin
alpha::vector< T >, 198
brushItems
Shape, 149

capacity
alpha::vector< T >, 199
childItems
Shape, 149
CIRCLE
all_shapes.h, 206
Circle, 25
Area, 30
Circle, 29
Draw, 30
getR, 31
isPointInside, 31
Perimeter, 32
setR, 32
setX, 32
setY, 33
ColumnEditDelegate, 34
ColumnEditDelegate, 35
createEditor, 35
setCanEdit, 35
const_iterator
alpha::vector< T >, 197
CreateBrushChild
Shape, 126
createEditor
ColumnEditDelegate, 35
CreateParentItem
Shape, 127
CreatePenChild
Shape, 128
CreatePointsChild
Shape, 129

deleteAllShapes
MainWindow, 62
RenderAreaManager, 115
ShapesManager, 153
deleteAllUsers
UserManager, 193
DeleteRenderAreaAll
ApiClient, 19
deleteShape
RenderAreaManager, 115
ShapesManager, 153
DeleteShapesAll
ApiClient, 19
DeleteTestimonialsAll
ApiClient, 19
deleteUser
UserManager, 193
DeleteUsersAll
ApiClient, 19
displayedTextChanged
MainWindow, 62
Draw
Circle, 30
Ellipse, 41
Line, 50
Polygon, 81
Polyline, 92
Rectangle, 102
Shape, 130
Square, 160
Text, 177
drawShapes
MainWindow, 62

ELLIPSE
all_shapes.h, 206
Ellipse, 36
Area, 41
Draw, 41
Ellipse, 40
getA, 42
getB, 42
isPointInside, 42
Perimeter, 43
setA, 43
setB, 43
setX, 44
setY, 44
end
alpha::vector< T >, 199
erase
alpha::vector< T >, 199

fromJson
Testimonial, 165

get_json_file
webservice.cpp, 261
getA
Ellipse, 42
getAuthor
Testimonial, 166
getB
Ellipse, 42
getBrush
Shape, 130
getBrushColor
Shape, 131
getBrushItems
Shape, 131
getBrushItemsEnd
Shape, 132
getBrushStyle
Shape, 132
getChildEnd

Shape, 132
getChildItems
Shape, 133
getContent
Testimonial, 166
getCurrUserRef
UserManager, 193
getDoNotShowAgain
TestimonialManager, 169
getEndPoint
Line, 51
getFont
Text, 178
getFontStyle
Text, 178
getFontWeight
Text, 179
getInstance
TestimonialManager, 170
getLength
Rectangle, 103
Square, 161
Text, 179
getPainter
Shape, 133
getParentItem
Shape, 134
getPassword
UserAccount, 186
getPen
Shape, 134
getPenCapStyle
Shape, 135
getPenColor
Shape, 135
getPenItems
Shape, 136
getPenItemsEnd
Shape, 136
getPenJoinStyle
Shape, 136
getPenStyle
Shape, 137
getPenWidth
Shape, 137
getPoints
Shape, 137
getPointsItems
Shape, 138
getPointsList
Polygon, 82
Polyline, 93
getR
Circle, 31
GetRenderArea
ApiClient, 19
getSatisfactoryTestimonials
TestimonialManager, 170
getSelected
Shape, 138
getShapeId
Shape, 139
GetShapes
ApiClient, 20
getShapes
RenderArea, 109
getShapeSelected
RenderArea, 109
getShapeSelectedIndex
RenderArea, 109
getShapesRef
RenderAreaManager, 116
ShapesManager, 153
getShapeType
Shape, 140
getStartPoint
Line, 51
GetTestimonials
ApiClient, 20
getTextAlignment
Text, 179
getTextColor
Text, 179
getTextString
Text, 180
getTimestamp
Testimonial, 166
getTrackerId
Shape, 140
getUsername
UserAccount, 186
GetUsers
ApiClient, 20
getWidth
Rectangle, 103
Text, 180
getX
Shape, 141
getY
Shape, 142
GoodDeleteReply
ApiClient, 20
GoodGetReply
ApiClient, 20
GoodPostReply
ApiClient, 21

hasUserGivenTestimonial
TestimonialManager, 170

insert
alpha::vector< T >, 199
isAdmin
UserAccount, 187
isGuest
Testimonial, 166
isPointInside

Circle, 31
Ellipse, 42
Line, 52
Polygon, 83
Polyline, 93
Rectangle, 103
Shape, 143
Square, 161
Text, 180
isSatisfactory
Testimonial, 166
iterator
alpha::vector< T >, 197

JsonToShapes
Parser, 73
JsonToTestimonials
Parser, 73
JsonToUsers
Parser, 73

LINE
all_shapes.h, 206
Line, 45
Area, 50
Draw, 50
getEndPoint, 51
getStartPoint, 51
isPointInside, 52
Line, 49
Move, 52
Perimeter, 53
setEndPoint, 53
setStartPoint, 54
setX, 54
setY, 55
loadAllData
AppDriver, 24
loadShapes
RenderAreaManager, 116
ShapesManager, 154
loadUsers
UserManager, 193
loginAttempt
MainWindow, 62
loginFailed
MainWindow, 62
loginRequested
LoginWindow, 57
loginSuccess
MainWindow, 63
LoginWindow, 56
loginRequested, 57
LoginWindow, 57
password, 57
signupRequested, 58
username, 58

main
main.cpp, 219
webservice.cpp, 262
main.cpp
main, 219
MainWindow, 59
~MainWindow, 62
deleteAllShapes, 62
displayedTextChanged, 62
drawShapes, 62
loginAttempt, 62
loginFailed, 62
loginSuccess, 63
MainWindow, 61
newUserAdded, 63
onLoginClicked, 63
onLoginRequest, 64
onRenderAreaChanged, 64
onRenderAreaNotChanged, 65
onSignupRequest, 65
onUserAuthentication, 66
onUserAuthenticationFailure, 67
shapeAdded, 68
shapeChanged, 68
shapeDeleted, 68
shapes_to_treeWidget, 68
showRenderStatusMessage, 70
moc_ApiClient.cpp
Q_CONSTINIT, 224
moc_UserManager.cpp
Q_CONSTINIT, 224
modifyDisplayedText
RenderAreaManager, 116
modifyShape
RenderAreaManager, 117
ShapesManager, 154
modifyUser
UserManager, 193
mouseDoubleClickEvent
RenderArea, 109
mouseMoveEvent
RenderArea, 109
mousePressEvent
RenderArea, 109
mouseReleaseEvent
RenderArea, 110
Move
Line, 52
Polygon, 83
Polyline, 94
Shape, 144

newUserAdded
MainWindow, 63
nextTracker
Shape, 149

onLoginClicked
MainWindow, 63
onLoginRequest

MainWindow, 64
onRenderAreaChanged
MainWindow, 64
onRenderAreaNotChanged
MainWindow, 65
onSignupRequest
MainWindow, 65
onUserAuthentication
MainWindow, 66
onUserAuthenticationFailure
MainWindow, 67
operator<
Shape, 148
shape.cpp, 236
operator=
alpha::vector< T >, 200
Parser, 74
UserAccount, 187
operator==
Shape, 148
shape.cpp, 236
operator[]
alpha::vector< T >, 200, 201

paintEvent
RenderArea, 110
parentItem
Shape, 149
Parser, 70
~Parser, 72
JsonToShapes, 73
JsonToTestimonials, 73
JsonToUsers, 73
operator=, 74
Parser, 72
PrintShapeVector, 74
ShapesToJson, 75
TestimonialsToJson, 75
UsersToJson, 75
password
LoginWindow, 57
penItems
Shape, 149
Perimeter
Circle, 32
Ellipse, 43
Line, 53
Polygon, 84
Polyline, 94
Rectangle, 104
Shape, 144
Square, 162
Text, 180
PI
shape.h, 238
pointsItems
Shape, 149
POLYGON
all_shapes.h, 206
Polygon, 76
Area, 81
Draw, 81
getPointsList, 82
isPointInside, 83
Move, 83
Perimeter, 84
Polygon, 80
setPointsList, 85
setX, 85
setY, 86
POLYLINE
all_shapes.h, 206
Polyline, 87
Area, 92
Draw, 92
getPointsList, 93
isPointInside, 93
Move, 94
Perimeter, 94
Polyline, 91
setPointsList, 95
setX, 95
setY, 96
PostRenderArea
ApiClient, 21
PostShapes
ApiClient, 21
PostTestimonials
ApiClient, 22
PostUsers
ApiClient, 22
PrintShapeVector
Parser, 74
push_back
alpha::vector< T >, 201

Q_CONSTINIT
moc_ApiClient.cpp, 224
moc_UserManager.cpp, 224
QT_WARNING_DISABLE_DEPRECATED, 13
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManag
97
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_
97

RECTANGLE
all_shapes.h, 206
Rectangle, 98
Area, 102
Draw, 102
getLength, 103
getWidth, 103
isPointInside, 103
Perimeter, 104
Rectangle, 101
setLength, 104
setWidth, 104
setX, 105

setY, 105
RenderArea, 106
getShapes, 109
getShapeSelected, 109
getShapeSelectedIndex, 109
mouseDoubleClickEvent, 109
mouseMoveEvent, 109
mousePressEvent, 109
mouseReleaseEvent, 110
paintEvent, 110
RenderArea, 109
resetSelection, 110
setEditPrivileges, 110
setRenderShapes, 111
setShapeSelectedIndex, 111
updateShapeDisplayCoords, 111
renderAreaChanged
RenderAreaManager, 118
RenderAreaManager, 112
∼RenderAreaManager, 114
addShape, 115
deleteAllShapes, 115
deleteShape, 115
getShapesRef, 116
loadShapes, 116
modifyDisplayedText, 116
modifyShape, 117
renderAreaChanged, 118
RenderAreaManager, 114
renderAreaNotChanged, 119
saveShapes, 119
statusMessage, 120
renderAreaNotChanged
RenderAreaManager, 119
reserve
alpha::vector< T >, 201
resetSelection
RenderArea, 110
resize
alpha::vector< T >, 202
run
AppDriver, 24

saveShapes
RenderAreaManager, 119
ShapesManager, 154
saveUsers
UserManager, 194
setA
Ellipse, 43
setAdmin
UserAccount, 188
setAlignment
Text, 181
setB
Ellipse, 43
setBrush
Shape, 145
setCanEdit
ColumnEditDelegate, 35
setDoNotShowAgain
TestimonialManager, 170
setEditPrivileges
RenderArea, 110
setEndPoint
Line, 53
setInternalBrush
Shape, 145
setInternalFont
Text, 181
setInternalPen
Shape, 145
setIsSatisfactory
Testimonial, 166
setLength
Rectangle, 104
Square, 162
Text, 181
setPassword
UserAccount, 188
setPen
Shape, 145
setPointsList
Polygon, 85
Polyline, 95
setR
Circle, 32
setRenderShapes
RenderArea, 111
setSelected
Shape, 146
setShapeId
Shape, 146
setShapeSelectedIndex
RenderArea, 111
setShapeType
Shape, 146
setStartPoint
Line, 54
setText
Text, 182
setTrackerId
Shape, 147
setUserAccount
UserAccount, 188
setUsername
UserAccount, 188
setWidth
Rectangle, 104
Text, 182
setX
Circle, 32
Ellipse, 44
Line, 54
Polygon, 85
Polyline, 95
Rectangle, 105

Shape, 147
Square, 162
Text, 182
setY
Circle, 33
Ellipse, 44
Line, 55
Polygon, 86
Polyline, 96
Rectangle, 105
Shape, 147
Square, 163
Text, 183
Shape, 120
~Shape, 125
allocateTrackerId, 126
Area, 126
brushItems, 149
childItems, 149
CreateBrushChild, 126
CreateParentItem, 127
CreatePenChild, 128
CreatePointsChild, 129
Draw, 130
getBrush, 130
getBrushColor, 131
getBrushItems, 131
getBrushItemsEnd, 132
getBrushStyle, 132
getChildEnd, 132
getChildItems, 133
getPainter, 133
getParentItem, 134
getPen, 134
getPenCapStyle, 135
getPenColor, 135
getPenItems, 136
getPenItemsEnd, 136
getPenJoinStyle, 136
getPenStyle, 137
getPenWidth, 137
getPoints, 137
getPointsItems, 138
getSelected, 138
getShapeId, 139
getShapeType, 140
getTrackerId, 140
getX, 141
getY, 142
isPointInside, 143
Move, 144
nextTracker, 149
operator<, 148
operator==, 148
parentItem, 149
penItems, 149
Perimeter, 144
pointsItems, 149
setBrush, 145
setInternalBrush, 145
setInternalPen, 145
setPen, 145
setSelected, 146
setShapeId, 146
setShapeType, 146
setTrackerId, 147
setX, 147
setY, 147
Shape, 124
trackersInUse, 149
shape.cpp
operator<, 236
operator==, 236
shape.h
PI, 238
shapeAdded
MainWindow, 68
shapeChanged
MainWindow, 68
shapeDeleted
MainWindow, 68
ShapeIDs
all_shapes.h, 205
shapes_to_treeWidget
MainWindow, 68
shapesChanged
ShapesManager, 154
ShapesManager, 150
~ShapesManager, 152
addShape, 152
deleteAllShapes, 153
deleteShape, 153
getShapesRef, 153
loadShapes, 154
modifyShape, 154
saveShapes, 154
shapesChanged, 154
ShapesManager, 152
shapesNotChanged, 155
statusMessage, 155
shapesNotChanged
ShapesManager, 155
ShapesToJson
Parser, 75
shouldPromptForTestimonial
TestimonialManager, 170
showRenderStatusMessage
MainWindow, 70
shutdown
AppDriver, 25
signupRequested
LoginWindow, 58
size
alpha::vector< T >, 202
SQUARE
all_shapes.h, 206

Square, 156
    Area, 160
    Draw, 160
    getLength, 161
    isPointInside, 161
    Perimeter, 162
    setLength, 162
    setX, 162
    setY, 163
    Square, 159
src/code/all_shapes.h, 205, 206
src/code/ApiClient.cpp, 206
src/code/ApiClient.h, 206, 207
src/code/AppDriver.cpp, 209
src/code/AppDriver.h, 209, 210
src/code/circle.cpp, 211
src/code/circle.h, 211, 212
src/code/ColumnEditDelegate.h, 212, 213
src/code/ellipse.cpp, 214
src/code/ellipse.h, 214, 215
src/code/line.cpp, 215
src/code/line.h, 216
src/code/loginwindow.cpp, 217
src/code/loginwindow.h, 217, 218
src/code/main.cpp, 219
src/code/mainwindow.cpp, 220
src/code/mainwindow.h, 220, 221
src/code/moc_ApiClient.cpp, 223
src/code/moc_UserManager.cpp, 224
src/code/Parser.cpp, 225
src/code/Parser.h, 225, 226
src/code/polygon.cpp, 227
src/code/polygon.h, 228
src/code/polyline.cpp, 229
src/code/polyline.h, 229, 230
src/code/rectangle.cpp, 231
src/code/rectangle.h, 231, 232
src/code/renderarea.cpp, 232
src/code/renderarea.h, 232, 233
src/code/RenderAreaManager.cpp, 234
src/code/RenderAreaManager.h, 234, 235
src/code/shape.cpp, 236
src/code/shape.h, 237, 238
src/code/ShapesManager.cpp, 239
src/code/ShapesManager.h, 240, 241
src/code/square.cpp, 241
src/code/square.h, 242
src/code/Testimonial.cpp, 243
src/code/Testimonial.h, 243, 244
src/code/TestimonialDialog.cpp, 244
src/code/TestimonialDialog.h, 245, 246
src/code/TestimonialManager.cpp, 246
src/code/TestimonialManager.h, 246, 247
src/code/TestimonialsDisplayDialog.cpp, 248
src/code/TestimonialsDisplayDialog.h, 248, 249
src/code/text.cpp, 250
src/code/text.h, 250, 251
src/code/UserAccount.cpp, 252
src/code/UserAccount.h, 252, 253
src/code/UserManager.cpp, 253
src/code/UserManager.h, 254, 255
src/code/vector.h, 255, 256
src/code/webservice.cpp, 260
startTrackingTime
    TestimonialManager, 170
statusMessage
    RenderAreaManager, 120
    ShapesManager, 155
    UserManager, 194
stopTrackingTime
    TestimonialManager, 170

Testimonial, 164
    fromJson, 165
    getAuthor, 166
    getContent, 166
    getTimestamp, 166
    isGuest, 166
    isSatisfactory, 166
    setIsSatisfactory, 166
    Testimonial, 165
    toJson, 166
TestimonialDialog, 167
    TestimonialDialog, 168
TestimonialManager, 168
    addTestimonial, 169
    getDoNotShowAgain, 169
    getInstance, 170
    getSatisfactoryTestimonials, 170
    hasUserGivenTestimonial, 170
    setDoNotShowAgain, 170
    shouldPromptForTestimonial, 170
    startTrackingTime, 170
    stopTrackingTime, 170
TestimonialsDisplayDialog, 171
    TestimonialsDisplayDialog, 172
TestimonialsToJson
    Parser, 75
TEXT
    all_shapes.h, 206
Text, 172
    Area, 177
    Draw, 177
    getFont, 178
    getFontStyle, 178
    getFontWeight, 179
    getLength, 179
    getTextAlignment, 179
    getTextColor, 179
    getTextString, 180
    getWidth, 180
    isPointInside, 180
    Perimeter, 180
    setAlignment, 181
    setInternalFont, 181
    setLength, 181
    setText, 182

setWidth, 182
setX, 182
setY, 183
Text, 177
toJson
Testimonial, 166
trackersInUse
Shape, 149

Ui, 13
updateShapeDisplayCoords
RenderArea, 111
UserAccount, 184
~UserAccount, 185
getPassword, 186
getUsername, 186
isAdmin, 187
operator=, 187
setAdmin, 188
setPassword, 188
setUserAccount, 188
setUsername, 188
UserAccount, 185, 186
userAuthenticated
UserManager, 194
userChanged
UserManager, 195
UserManager, 189
~UserManager, 191
addUser, 192
authenticate, 192
authenticationFailed, 192
deleteAllUsers, 193
deleteUser, 193
getCurrUserRef, 193
loadUsers, 193
modifyUser, 193
saveUsers, 194
statusMessage, 194
userAuthenticated, 194
userChanged, 195
UserManager, 191
userNotChanged, 195
username
LoginWindow, 58
userNotChanged
UserManager, 195
UsersToJson
Parser, 75

vector
alpha::vector< T >, 197, 198
vector.h
ALPHA_VECTOR_H, 256

webservice.cpp
get_json_file, 261
main, 262