2D Graphics Modeler - Team Alphawolves

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 . . . 19
8.1.2 Constructor & Destructor Documentation . . . 20
8.1.2.1 ApiClient() . . . 20
8.1.3 Member Function Documentation . . . 20
8.1.3.1 AnalyzeDeleteReply . . . 20
8.1.3.2 AnalyzeGetReply . . . 21
8.1.3.3 AnalyzePostReply . . . 22
8.1.3.4 BadDeleteReply . . . 23
8.1.3.5 BadGetReply . . . 23
8.1.3.6 BadPostReply . . . 24
8.1.3.7 DeleteRenderAreaAll() . . . 24
8.1.3.8 DeleteShapesAll() . . . 25
8.1.3.9 DeleteTestimonialsAll() . . . 25
8.1.3.10 DeleteUsersAll() . . . 25
8.1.3.11 GetRenderArea() . . . 26
8.1.3.12 GetShapes() . . . 26
8.1.3.13 GetTestimonials() . . . 26
8.1.3.14 GetUsers() . . . 27
8.1.3.15 GoodDeleteReply . . . 27
8.1.3.16 GoodGetReply . . . 27

8.1.3.17 GoodPostReply . . . 28
8.1.3.18 PostRenderArea() . . . 28
8.1.3.19 PostShapes() . . . 29
8.1.3.20 PostTestimonials() . . . 29
8.1.3.21 PostUsers() . . . 29
8.1.4 Member Data Documentation . . . 30
8.1.4.1 manager . . . 30
8.2 AppDriver Class Reference . . . 30
8.2.1 Detailed Description . . . 34
8.2.2 Constructor & Destructor Documentation . . . 34
8.2.2.1 AppDriver() . . . 34
8.2.2.2 ~AppDriver() . . . 34
8.2.3 Member Function Documentation . . . 35
8.2.3.1 connectFrontendToDriver() . . . 35
8.2.3.2 connectManagersToFrontend() . . . 36
8.2.3.3 loadAllData() . . . 37
8.2.3.4 onDeleteAllUsers . . . 37
8.2.3.5 onLoginAttempt . . . 37
8.2.3.6 onNewUser . . . 38
8.2.3.7 onRenderDeleteAllShapes . . . 38
8.2.3.8 onRenderShapeAdded . . . 38
8.2.3.9 onRenderShapeChanged . . . 39
8.2.3.10 onRenderShapeDeleted . . . 39
8.2.3.11 onUserDeleted . . . 39
8.2.3.12 onUserModified . . . 39
8.2.3.13 run() . . . 40
8.2.3.14 shutdown() . . . 41
8.2.4 Member Data Documentation . . . 41
8.2.4.1 mainWindow . . . 41
8.2.4.2 renderedShapes . . . 41
8.2.4.3 user . . . 41
8.3 Circle Class Reference . . . 42
8.3.1 Detailed Description . . . 45
8.3.2 Constructor & Destructor Documentation . . . 45
8.3.2.1 Circle() . . . 45
8.3.3 Member Function Documentation . . . 46
8.3.3.1 Area() . . . 46
8.3.3.2 Draw() . . . 46
8.3.3.3 getR() . . . 47
8.3.3.4 isPointInside() . . . 48
8.3.3.5 Perimeter() . . . 48
8.3.3.6 setR() . . . 49

8.3.3.7 setX() . . . 49
8.3.3.8 setY() . . . 50
8.3.4 Member Data Documentation . . . 50
8.3.4.1 r . . . 50
8.4 ColumnEditDelegate Class Reference . . . 51
8.4.1 Detailed Description . . . 52
8.4.2 Constructor & Destructor Documentation . . . 53
8.4.2.1 ColumnEditDelegate() . . . 53
8.4.3 Member Function Documentation . . . 54
8.4.3.1 createEditor() . . . 54
8.4.3.2 setCanEdit() . . . 54
8.4.4 Member Data Documentation . . . 54
8.4.4.1 canEdit . . . 54
8.5 Ellipse Class Reference . . . 55
8.5.1 Detailed Description . . . 58
8.5.2 Constructor & Destructor Documentation . . . 59
8.5.2.1 Ellipse() . . . 59
8.5.3 Member Function Documentation . . . 59
8.5.3.1 Area() . . . 59
8.5.3.2 Draw() . . . 60
8.5.3.3 getA() . . . 60
8.5.3.4 getB() . . . 61
8.5.3.5 isPointInside() . . . 61
8.5.3.6 Perimeter() . . . 62
8.5.3.7 setA() . . . 62
8.5.3.8 setB() . . . 62
8.5.3.9 setX() . . . 63
8.5.3.10 setY() . . . 63
8.5.4 Member Data Documentation . . . 64
8.5.4.1 a . . . 64
8.5.4.2 b . . . 64
8.6 Line Class Reference . . . 64
8.6.1 Detailed Description . . . 68
8.6.2 Constructor & Destructor Documentation . . . 69
8.6.2.1 Line() . . . 69
8.6.3 Member Function Documentation . . . 69
8.6.3.1 Area() . . . 69
8.6.3.2 Draw() . . . 70
8.6.3.3 getEndPoint() . . . 70
8.6.3.4 getStartPoint() . . . 71
8.6.3.5 isPointInside() . . . 71
8.6.3.6 Move() . . . 71

8.6.3.7 Perimeter() . . . 72
8.6.3.8 setEndPoint() . . . 73
8.6.3.9 setStartPoint() . . . 73
8.6.3.10 setX() . . . 73
8.6.3.11 setY() . . . 74
8.6.4 Member Data Documentation . . . 74
8.6.4.1 endPoint . . . 74
8.6.4.2 startPoint . . . 75
8.7 LoginWindow Class Reference . . . 75
8.7.1 Detailed Description . . . 79
8.7.2 Constructor & Destructor Documentation . . . 79
8.7.2.1 LoginWindow() . . . 79
8.7.3 Member Function Documentation . . . 79
8.7.3.1 attemptLogin . . . 79
8.7.3.2 attemptSignup . . . 80
8.7.3.3 loginRequested . . . 80
8.7.3.4 password() . . . 81
8.7.3.5 showLoginPage . . . 81
8.7.3.6 showSignupPage . . . 82
8.7.3.7 signupRequested . . . 82
8.7.3.8 username() . . . 83
8.7.4 Member Data Documentation . . . 83
8.7.4.1 backToLoginBtn . . . 83
8.7.4.2 loginBtn . . . 83
8.7.4.3 loginPage . . . 83
8.7.4.4 loginPassEdit . . . 83
8.7.4.5 loginUserEdit . . . 83
8.7.4.6 signupBtn . . . 84
8.7.4.7 signupPage . . . 84
8.7.4.8 signupPassEdit . . . 84
8.7.4.9 signupUserEdit . . . 84
8.7.4.10 stack . . . 84
8.7.4.11 toSignupBtn . . . 84
8.8 MainWindow Class Reference . . . 85
8.8.1 Detailed Description . . . 92
8.8.2 Constructor & Destructor Documentation . . . 92
8.8.2.1 MainWindow() [1/2] . . . 92
8.8.2.2 MainWindow() [2/2] . . . 93
8.8.2.3 ~MainWindow() . . . 93
8.8.3 Member Function Documentation . . . 94
8.8.3.1 addToShapeTree() . . . 94
8.8.3.2 createAlignmentComboBox() . . . 95

8.8.3.3 createBrushStyleComboBox() . . . 96
8.8.3.4 createColorComboBox() . . . 96
8.8.3.5 createFontComboBox() . . . 97
8.8.3.6 createFontStyleComboBox() . . . 97
8.8.3.7 createFontWeightComboBox() . . . 98
8.8.3.8 createPenCapStyleComboBox() . . . 98
8.8.3.9 createPenJoinStyleComboBox() . . . 99
8.8.3.10 createPenStyleComboBox() . . . 99
8.8.3.11 createPenWidthSpinBox() . . . 100
8.8.3.12 createShapeTableTab() . . . 100
8.8.3.13 createShapeTypeComboBox() . . . 101
8.8.3.14 deleteAllShapes . . . 101
8.8.3.15 displayedTextChanged . . . 102
8.8.3.16 drawShapes() . . . 102
8.8.3.17 loadStyleSheet() . . . 102
8.8.3.18 loginAttempt . . . 102
8.8.3.19 loginFailed . . . 103
8.8.3.20 loginSuccess . . . 103
8.8.3.21 newUserAdded . . . 103
8.8.3.22 on_actionnew_circle_button_triggered . . . 104
8.8.3.23 on_actionnew_ellipse_button_triggered . . . 104
8.8.3.24 on_actionnew_line_button_triggered . . . 105
8.8.3.25 on_actionnew_polygon_button_triggered . . . 106
8.8.3.26 on_actionnew_polyline_button_triggered . . . 107
8.8.3.27 on_actionnew_rectange_button_triggered . . . 108
8.8.3.28 on_actionnew_square_button_triggered . . . 109
8.8.3.29 on_actionnew_text_button_triggered . . . 110
8.8.3.30 on_actionremove_shape_button_triggered . . . 111
8.8.3.31 onComboBoxChanged . . . 112
8.8.3.32 onContactUsClicked . . . 113
8.8.3.33 onLoginClicked . . . 113
8.8.3.34 onLoginRequest . . . 114
8.8.3.35 onRenderAreaChanged . . . 114
8.8.3.36 onRenderAreaNotChanged . . . 116
8.8.3.37 onSignupRequest . . . 116
8.8.3.38 onSortMethodChanged . . . 117
8.8.3.39 onSpinBoxChanged . . . 117
8.8.3.40 onToggleStyle . . . 118
8.8.3.41 onTreeWidgetItemChanged . . . 119
8.8.3.42 onUserAuthentication . . . 119
8.8.3.43 onUserAuthenticationFailure . . . 121
8.8.3.44 populateShapeTable() . . . 121

8.8.3.45 selection_sort() . . . 122
8.8.3.46 setupTestimonials() . . . 122
8.8.3.47 shapeAdded . . . 123
8.8.3.48 shapeChanged . . . 124
8.8.3.49 shapeDeleted . . . 125
8.8.3.50 shapes_to_treeWidget() . . . 125
8.8.3.51 showRenderStatusMessage . . . 127
8.8.3.52 showTestimonialPrompt . . . 127
8.8.3.53 showTestimonialsDisplay . . . 128
8.8.3.54 sortByArea() . . . 128
8.8.3.55 sortById() . . . 129
8.8.3.56 sortByPerimeter() . . . 129
8.8.4 Member Data Documentation . . . 130
8.8.4.1 contactUsWidget . . . 130
8.8.4.2 contactWindow . . . 130
8.8.4.3 currUser . . . 130
8.8.4.4 delegate . . . 130
8.8.4.5 logoLabel . . . 130
8.8.4.6 renderArea . . . 130
8.8.4.7 renderShapes . . . 130
8.8.4.8 shapeTable . . . 131
8.8.4.9 sortDropdown . . . 131
8.8.4.10 sortOrderDropdown . . . 131
8.8.4.11 tabWidget . . . 131
8.8.4.12 teamNameLabel . . . 131
8.8.4.13 ui . . . 131
8.8.4.14 userStatusLabel . . . 131
8.9 Parser::MorphicShape Struct Reference . . . 132
8.9.1 Detailed Description . . . 133
8.9.2 Member Data Documentation . . . 133
8.9.2.1 brush . . . 133
8.9.2.2 coords . . . 133
8.9.2.3 font . . . 133
8.9.2.4 pen . . . 134
8.9.2.5 shapeDimensions . . . 134
8.9.2.6 shapeId . . . 134
8.9.2.7 shapeType . . . 134
8.9.2.8 textAlignment . . . 134
8.9.2.9 textColor . . . 134
8.9.2.10 textString . . . 134
8.9.2.11 trackerId . . . 135
8.10 Parser Class Reference . . . 135

8.10.1 Detailed Description . . . 138
8.10.2 Constructor & Destructor Documentation . . . 139
8.10.2.1 Parser() [1/3] . . . 139
8.10.2.2 ~Parser() . . . 139
8.10.2.3 Parser() [2/3] . . . 139
8.10.2.4 Parser() [3/3] . . . 140
8.10.3 Member Function Documentation . . . 140
8.10.3.1 AppendBrushData() . . . 140
8.10.3.2 AppendCommonShapeData() . . . 141
8.10.3.3 AppendTextData() . . . 143
8.10.3.4 BuildShape() . . . 144
8.10.3.5 ExtractArray() . . . 145
8.10.3.6 ExtractInteger() . . . 145
8.10.3.7 ExtractKey() . . . 146
8.10.3.8 ExtractLiteral() . . . 146
8.10.3.9 ExtractValue() . . . 147
8.10.3.10 GetAlignmentFlag() . . . 148
8.10.3.11 GetBrushStyle() . . . 149
8.10.3.12 GetColor() . . . 150
8.10.3.13 GetFontStyle() . . . 151
8.10.3.14 GetFontWeight() . . . 152
8.10.3.15 GetPenCapStyle() . . . 152
8.10.3.16 GetPenJoinStyle() . . . 153
8.10.3.17 GetPenStyle() . . . 154
8.10.3.18 GetShapeDimensions() . . . 155
8.10.3.19 JsonToShapes() . . . 157
8.10.3.20 JsonToTestimonials() . . . 157
8.10.3.21 JsonToUsers() . . . 158
8.10.3.22 operator=() [1/2] . . . 159
8.10.3.23 operator=() [2/2] . . . 159
8.10.3.24 ParseJsonObject() . . . 160
8.10.3.25 PrintShapeVector() . . . 161
8.10.3.26 ShapesToJson() . . . 161
8.10.3.27 SkipWhitespace() . . . 162
8.10.3.28 StringToVector() . . . 163
8.10.3.29 TestimonialsToJson() . . . 164
8.10.3.30 UpdateAccumulator() . . . 164
8.10.3.31 UpdateUserAccumulator() . . . 165
8.10.3.32 UsersToJson() . . . 166
8.11 Polygon Class Reference . . . 166
8.11.1 Detailed Description . . . 170
8.11.2 Constructor & Destructor Documentation . . . 170

8.11.2.1 Polygon() . . . 170
8.11.3 Member Function Documentation . . . 171
8.11.3.1 Area() . . . 171
8.11.3.2 Draw() . . . 172
8.11.3.3 getPointsList() . . . 172
8.11.3.4 isPointInside() . . . 173
8.11.3.5 Move() . . . 173
8.11.3.6 Perimeter() . . . 174
8.11.3.7 setPointsList() . . . 175
8.11.3.8 setX() . . . 175
8.11.3.9 setY() . . . 176
8.11.4 Member Data Documentation . . . 176
8.11.4.1 pointsList . . . 176
8.12 Polyline Class Reference . . . 177
8.12.1 Detailed Description . . . 180
8.12.2 Constructor & Destructor Documentation . . . 180
8.12.2.1 Polyline() . . . 180
8.12.3 Member Function Documentation . . . 181
8.12.3.1 Area() . . . 181
8.12.3.2 Draw() . . . 181
8.12.3.3 getPointsList() . . . 182
8.12.3.4 isPointInside() . . . 183
8.12.3.5 Move() . . . 183
8.12.3.6 Perimeter() . . . 184
8.12.3.7 setPointsList() . . . 184
8.12.3.8 setX() . . . 185
8.12.3.9 setY() . . . 185
8.12.4 Member Data Documentation . . . 186
8.12.4.1 pointsList . . . 186
8.13 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t Struct Reference . 186
8.14 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t Struct Reference . . . 187
8.15 Parser::RawUser Struct Reference . . . 187
8.15.1 Detailed Description . . . 188
8.15.2 Member Data Documentation . . . 188
8.15.2.1 admin . . . 188
8.15.2.2 hasAdmin . . . 188
8.15.2.3 hasPassword . . . 188
8.15.2.4 hasUsername . . . 188
8.15.2.5 password . . . 189
8.15.2.6 username . . . 189
8.16 Rectangle Class Reference . . . 189
8.16.1 Detailed Description . . . 193

8.16.2 Constructor & Destructor Documentation . . . 194
8.16.2.1 Rectangle() . . . 194
8.16.3 Member Function Documentation . . . 194
8.16.3.1 Area() . . . 194
8.16.3.2 Draw() . . . 195
8.16.3.3 getLength() . . . 195
8.16.3.4 getWidth() . . . 196
8.16.3.5 isPointInside() . . . 196
8.16.3.6 Perimeter() . . . 196
8.16.3.7 setLength() . . . 197
8.16.3.8 setWidth() . . . 197
8.16.3.9 setX() . . . 197
8.16.3.10 setY() . . . 198
8.16.4 Member Data Documentation . . . 198
8.16.4.1 length . . . 198
8.16.4.2 width . . . 199
8.17 RenderArea Class Reference . . . 199
8.17.1 Detailed Description . . . 203
8.17.2 Constructor & Destructor Documentation . . . 203
8.17.2.1 RenderArea() . . . 203
8.17.3 Member Function Documentation . . . 203
8.17.3.1 getShapes() . . . 203
8.17.3.2 getShapeSelected() . . . 203
8.17.3.3 getShapeSelectedIndex() . . . 203
8.17.3.4 mouseDoubleClickEvent() . . . 203
8.17.3.5 mouseMoveEvent() . . . 204
8.17.3.6 mousePressEvent() . . . 204
8.17.3.7 mouseReleaseEvent() . . . 204
8.17.3.8 paintEvent() . . . 205
8.17.3.9 resetSelection() . . . 205
8.17.3.10 setEditPrivileges() . . . 205
8.17.3.11 setRenderShapes() . . . 205
8.17.3.12 setShapeSelectedIndex() . . . 205
8.17.3.13 updateShapeDisplayCoords() . . . 206
8.17.4 Member Data Documentation . . . 206
8.17.4.1 allowEditing . . . 206
8.17.4.2 renderShapes . . . 207
8.17.4.3 shapeSelectedIndex . . . 207
8.18 RenderAreaManager Class Reference . . . 207
8.18.1 Detailed Description . . . 211
8.18.2 Constructor & Destructor Documentation . . . 211
8.18.2.1 RenderAreaManager() . . . 211

8.18.2.2 ~RenderAreaManager() . . . 212
8.18.3 Member Function Documentation . . . 212
8.18.3.1 addShape() . . . 212
8.18.3.2 deleteAllShapes() . . . 213
8.18.3.3 deleteShape() . . . 213
8.18.3.4 getShapesRef() . . . 214
8.18.3.5 loadShapes() . . . 214
8.18.3.6 modifyDisplayedText() . . . 214
8.18.3.7 modifyShape() . . . 215
8.18.3.8 onBadDeleteResponse . . . 216
8.18.3.9 onBadGetResponse . . . 217
8.18.3.10 onBadPostResponse . . . 218
8.18.3.11 onGoodDeleteResponse . . . 218
8.18.3.12 onGoodGetResponse . . . 219
8.18.3.13 onGoodPostResponse . . . 220
8.18.3.14 renderAreaChanged . . . 220
8.18.3.15 renderAreaNotChanged . . . 221
8.18.3.16 saveShapes() . . . 221
8.18.3.17 statusMessage . . . 222
8.18.4 Member Data Documentation . . . 222
8.18.4.1 client . . . 222
8.18.4.2 parse . . . 223
8.18.4.3 renderedShapes . . . 223
8.19 Shape Interface Reference . . . 223
8.19.1 Detailed Description . . . 227
8.19.2 Constructor & Destructor Documentation . . . 227
8.19.2.1 Shape() [1/2] . . . 227
8.19.2.2 ~Shape() . . . 228
8.19.2.3 Shape() [2/2] . . . 229
8.19.3 Member Function Documentation . . . 229
8.19.3.1 allocateTrackerId() . . . 229
8.19.3.2 Area() . . . 230
8.19.3.3 CreateBrushChild() . . . 230
8.19.3.4 CreateParentItem() . . . 231
8.19.3.5 CreatePenChild() . . . 232
8.19.3.6 CreatePointsChild() . . . 232
8.19.3.7 Draw() . . . 233
8.19.3.8 getBrush() . . . 234
8.19.3.9 getBrushColor() . . . 234
8.19.3.10 getBrushItems() . . . 235
8.19.3.11 getBrushItemsEnd() . . . 235
8.19.3.12 getBrushStyle() . . . 236

8.19.3.13 getChildEnd() . . . 236
8.19.3.14 getChildItems() . . . 237
8.19.3.15 getPainter() . . . 237
8.19.3.16 getParentItem() . . . 238
8.19.3.17 getPen() . . . 239
8.19.3.18 getPenCapStyle() . . . 239
8.19.3.19 getPenColor() . . . 240
8.19.3.20 getPenItems() . . . 240
8.19.3.21 getPenItemsEnd() . . . 240
8.19.3.22 getPenJoinStyle() . . . 241
8.19.3.23 getPenStyle() . . . 241
8.19.3.24 getPenWidth() . . . 242
8.19.3.25 getPoints() . . . 242
8.19.3.26 getPointsItems() . . . 243
8.19.3.27 getSelected() . . . 243
8.19.3.28 getShapeId() . . . 244
8.19.3.29 getShapeType() . . . 245
8.19.3.30 getTrackerId() . . . 245
8.19.3.31 getX() . . . 246
8.19.3.32 getY() . . . 247
8.19.3.33 isPointInside() . . . 248
8.19.3.34 Move() . . . 249
8.19.3.35 operator=() . . . 250
8.19.3.36 Perimeter() . . . 250
8.19.3.37 setBrush() . . . 250
8.19.3.38 setInternalBrush() . . . 250
8.19.3.39 setInternalPen() . . . 251
8.19.3.40 setPen() . . . 251
8.19.3.41 setSelected() . . . 251
8.19.3.42 setShapeType() . . . 251
8.19.3.43 setTrackerId() . . . 252
8.19.3.44 setX() . . . 252
8.19.3.45 setY() . . . 253
8.19.4 Friends And Related Symbol Documentation . . . 253
8.19.4.1 operator< . . . 253
8.19.4.2 operator== . . . 253
8.19.5 Member Data Documentation . . . 254
8.19.5.1 brush . . . 254
8.19.5.2 brushItems . . . 254
8.19.5.3 childItems . . . 254
8.19.5.4 coords . . . 254
8.19.5.5 isSelected . . . 254

8.19.5.6 nextTracker . . . 254
8.19.5.7 painter . . . 255
8.19.5.8 parentItem . . . 255
8.19.5.9 pen . . . 255
8.19.5.10 penItems . . . 255
8.19.5.11 pointsItems . . . 255
8.19.5.12 shapeId . . . 255
8.19.5.13 shapeType . . . 255
8.19.5.14 trackerId . . . 256
8.19.5.15 trackersInUse . . . 256
8.20 ShapesManager Class Reference . . . 256
8.20.1 Constructor & Destructor Documentation . . . 259
8.20.1.1 ShapesManager() . . . 259
8.20.1.2 ∼ShapesManager() . . . 260
8.20.2 Member Function Documentation . . . 260
8.20.2.1 addShape() . . . 260
8.20.2.2 deleteAllShapes() . . . 261
8.20.2.3 deleteShape() . . . 261
8.20.2.4 getShapesRef() . . . 262
8.20.2.5 loadShapes() . . . 262
8.20.2.6 modifyShape() . . . 262
8.20.2.7 onBadDeleteResponse . . . 262
8.20.2.8 onBadGetResponse . . . 263
8.20.2.9 onBadPostResponse . . . 263
8.20.2.10 onGoodDeleteResponse . . . 264
8.20.2.11 onGoodGetResponse . . . 264
8.20.2.12 onGoodPostResponse . . . 265
8.20.2.13 saveShapes() . . . 265
8.20.2.14 shapesChanged . . . 266
8.20.2.15 shapesNotChanged . . . 266
8.20.2.16 statusMessage . . . 267
8.20.3 Member Data Documentation . . . 267
8.20.3.1 client . . . 267
8.20.3.2 parse . . . 267
8.20.3.3 shapes . . . 268
8.21 Square Class Reference . . . 268
8.21.1 Detailed Description . . . 272
8.21.2 Constructor & Destructor Documentation . . . 272
8.21.2.1 Square() . . . 272
8.21.3 Member Function Documentation . . . 273
8.21.3.1 Area() . . . 273
8.21.3.2 Draw() . . . 273

8.21.3.3 getLength() . . . 274
8.21.3.4 isPointInside() . . . 274
8.21.3.5 Perimeter() . . . 275
8.21.3.6 setLength() . . . 275
8.21.3.7 setX() . . . 276
8.21.3.8 setY() . . . 276
8.21.4 Member Data Documentation . . . 277
8.21.4.1 length . . . 277
8.22 Testimonial Class Reference . . . 277
8.22.1 Detailed Description . . . 280
8.22.2 Constructor & Destructor Documentation . . . 280
8.22.2.1 Testimonial() . . . 280
8.22.3 Member Function Documentation . . . 280
8.22.3.1 fromJson() . . . 280
8.22.3.2 getAuthor() . . . 281
8.22.3.3 getContent() . . . 281
8.22.3.4 getTimestamp() . . . 281
8.22.3.5 isGuest() . . . 281
8.22.3.6 isSatisfactory() . . . 282
8.22.3.7 setIsSatisfactory() . . . 282
8.22.3.8 toJson() . . . 282
8.22.4 Member Data Documentation . . . 282
8.22.4.1 m_author . . . 282
8.22.4.2 m_content . . . 282
8.22.4.3 m_isGuest . . . 282
8.22.4.4 m_isSatisfactory . . . 282
8.22.4.5 m_timestamp . . . 283
8.23 TestimonialDialog Class Reference . . . 283
8.23.1 Detailed Description . . . 285
8.23.2 Constructor & Destructor Documentation . . . 285
8.23.2.1 TestimonialDialog() . . . 285
8.23.3 Member Function Documentation . . . 286
8.23.3.1 onCancel . . . 286
8.23.3.2 onSubmit . . . 286
8.23.4 Member Data Documentation . . . 287
8.23.4.1 m_authorEdit . . . 287
8.23.4.2 m_contentEdit . . . 287
8.23.4.3 m_doNotShowAgain . . . 288
8.24 TestimonialManager Class Reference . . . 288
8.24.1 Detailed Description . . . 292
8.24.2 Constructor & Destructor Documentation . . . 293
8.24.2.1 TestimonialManager() . . . 293

8.24.2.2 ~TestimonialManager() . . . 294
8.24.3 Member Function Documentation . . . 294
8.24.3.1 addTestimonial() . . . 294
8.24.3.2 checkTimeAndPrompt() . . . 295
8.24.3.3 getDoNotShowAgain() . . . 295
8.24.3.4 getInstance() . . . 296
8.24.3.5 getSatisfactoryTestimonials() . . . 296
8.24.3.6 hasUserGivenTestimonial() . . . 297
8.24.3.7 loadTestimonials() . . . 297
8.24.3.8 onBadGetResponse . . . 297
8.24.3.9 onBadPostResponse . . . 298
8.24.3.10 onGoodGetResponse . . . 298
8.24.3.11 onGoodPostResponse . . . 298
8.24.3.12 saveTestimonials() . . . 299
8.24.3.13 setDoNotShowAgain() . . . 299
8.24.3.14 shouldPromptForTestimonial . . . 300
8.24.3.15 startTrackingTime() . . . 300
8.24.3.16 stopTrackingTime() . . . 300
8.24.4 Member Data Documentation . . . 301
8.24.4.1 client . . . 301
8.24.4.2 INITIAL_PROMPT_TIME . . . 301
8.24.4.3 m_doNotShowAgain . . . 301
8.24.4.4 m_testimonials . . . 301
8.24.4.5 m_trackingTimer . . . 301
8.24.4.6 m_userTimeTracking . . . 301
8.24.4.7 parse . . . 301
8.24.4.8 REPEAT_PROMPT_TIME . . . 302
8.25 TestimonialsDisplayDialog Class Reference . . . 302
8.25.1 Detailed Description . . . 303
8.25.2 Constructor & Destructor Documentation . . . 303
8.25.2.1 TestimonialsDisplayDialog() . . . 303
8.25.3 Member Function Documentation . . . 304
8.25.3.1 refreshTestimonials() . . . 304
8.25.4 Member Data Documentation . . . 305
8.25.4.1 m_testimonialsLayout . . . 305
8.26 Text Class Reference . . . 305
8.26.1 Detailed Description . . . 310
8.26.2 Constructor & Destructor Documentation . . . 310
8.26.2.1 Text() . . . 310
8.26.3 Member Function Documentation . . . 311
8.26.3.1 Area() . . . 311
8.26.3.2 Draw() . . . 311

8.26.3.3 getFont() . . . 312
8.26.3.4 getFontStyle() . . . 313
8.26.3.5 getFontWeight() . . . 313
8.26.3.6 getLength() . . . 314
8.26.3.7 getTextAlignment() . . . 314
8.26.3.8 getTextColor() . . . 314
8.26.3.9 getTextString() . . . 315
8.26.3.10 getWidth() . . . 315
8.26.3.11 isPointInside() . . . 315
8.26.3.12 Perimeter() . . . 316
8.26.3.13 setAlignment() . . . 316
8.26.3.14 setInternalFont() . . . 317
8.26.3.15 setLength() . . . 317
8.26.3.16 setText() . . . 317
8.26.3.17 setWidth() . . . 318
8.26.3.18 setX() . . . 318
8.26.3.19 setY() . . . 319
8.26.4 Member Data Documentation . . . 319
8.26.4.1 font . . . 319
8.26.4.2 length . . . 319
8.26.4.3 textAlignment . . . 319
8.26.4.4 textColor . . . 320
8.26.4.5 textString . . . 320
8.26.4.6 width . . . 320
8.27 UserAccount Class Reference . . . 320
8.27.1 Detailed Description . . . 323
8.27.2 Constructor & Destructor Documentation . . . 323
8.27.2.1 UserAccount() [1/4] . . . 323
8.27.2.2 UserAccount() [2/4] . . . 323
8.27.2.3 ~UserAccount() . . . 323
8.27.2.4 UserAccount() [3/4] . . . 324
8.27.2.5 UserAccount() [4/4] . . . 324
8.27.3 Member Function Documentation . . . 324
8.27.3.1 getPassword() . . . 324
8.27.3.2 getUsername() . . . 325
8.27.3.3 isAdmin() . . . 325
8.27.3.4 operator=() [1/2] . . . 325
8.27.3.5 operator=() [2/2] . . . 326
8.27.3.6 setAdmin() . . . 326
8.27.3.7 setPassword() . . . 326
8.27.3.8 setUserAccount() . . . 326
8.27.3.9 setUsername() . . . 326

8.27.4 Member Data Documentation . . . 327
8.27.4.1 admin . . . 327
8.27.4.2 password . . . 327
8.27.4.3 username . . . 327
8.28 UserManager Class Reference . . . 327
8.28.1 Detailed Description . . . 331
8.28.2 Constructor & Destructor Documentation . . . 331
8.28.2.1 UserManager() . . . 331
8.28.2.2 ∼UserManager() . . . 332
8.28.3 Member Function Documentation . . . 332
8.28.3.1 addUser() . . . 332
8.28.3.2 authenticate() . . . 333
8.28.3.3 authenticationFailed . . . 334
8.28.3.4 deleteAllUsers() . . . 334
8.28.3.5 deleteUser() . . . 334
8.28.3.6 getCurrUserRef() . . . 334
8.28.3.7 loadUsers() . . . 335
8.28.3.8 modifyUser() . . . 335
8.28.3.9 onBadDeleteResponse . . . 335
8.28.3.10 onBadGetResponse . . . 336
8.28.3.11 onBadPostResponse . . . 336
8.28.3.12 onGoodDeleteResponse . . . 337
8.28.3.13 onGoodGetResponse . . . 338
8.28.3.14 onGoodPostResponse . . . 338
8.28.3.15 saveUsers() . . . 339
8.28.3.16 statusMessage . . . 339
8.28.3.17 userAuthenticated . . . 340
8.28.3.18 userChanged . . . 340
8.28.3.19 userNotChanged . . . 340
8.28.4 Member Data Documentation . . . 340
8.28.4.1 client . . . 340
8.28.4.2 currUser . . . 341
8.28.4.3 parse . . . 341
8.28.4.4 users . . . 341
8.29 alpha::vector< T > Class Template Reference . . . 341
8.29.1 Member Typedef Documentation . . . 344
8.29.1.1 const_iterator . . . 344
8.29.1.2 iterator . . . 344
8.29.2 Constructor & Destructor Documentation . . . 344
8.29.2.1 vector() [1/4] . . . 344
8.29.2.2 vector() [2/4] . . . 345
8.29.2.3 vector() [3/4] . . . 345

8.29.2.4 vector() [4/4] . . . 345
8.29.2.5 ~vector() . . . 345
8.29.3 Member Function Documentation . . . 346
8.29.3.1 begin() [1/2] . . . 346
8.29.3.2 begin() [2/2] . . . 346
8.29.3.3 capacity() . . . 346
8.29.3.4 end() [1/2] . . . 346
8.29.3.5 end() [2/2] . . . 346
8.29.3.6 erase() . . . 347
8.29.3.7 insert() . . . 347
8.29.3.8 operator=() [1/2] . . . 347
8.29.3.9 operator=() [2/2] . . . 348
8.29.3.10 operator[]() [1/2] . . . 348
8.29.3.11 operator[]() [2/2] . . . 348
8.29.3.12 push_back() . . . 348
8.29.3.13 reserve() . . . 349
8.29.3.14 resize() . . . 349
8.29.3.15 size() . . . 350
8.29.4 Member Data Documentation . . . 350
8.29.4.1 elem . . . 350
8.29.4.2 size_v . . . 350
8.29.4.3 space . . . 350

9 File Documentation **351**

9.1 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.cpp File Reference . . . 351
9.2 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.h File Reference . . . 351
9.2.1 Detailed Description . . . 352
9.3 ApiClient.h . . . 352
9.4 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.cpp File Reference . . . 353
9.5 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.h File Reference . . . 353
9.6 AppDriver.h . . . 354
9.7 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_ApiClient.cpp File Reference . . . 355
9.7.1 Macro Definition Documentation . . . 356
9.7.1.1 Q_CONSTINIT . . . 356
9.7.2 Variable Documentation . . . 356
9.7.2.1 qt_meta_data_ZN9ApiClientE . . . 356
9.8 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_User↩Manager.cpp File Reference . . . 356
9.8.1 Macro Definition Documentation . . . 357

9.8.1.1 Q_CONSTINIT . . . 357
9.8.2 Variable Documentation . . . 357
9.8.2.1 qt_meta_data_ZN11UserManagerE . . . 357
9.9 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.cpp File Reference . . . 357
9.10 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h File Reference . . . 357
9.11 Parser.h . . . 358
9.12 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.cpp File Reference . . . 360
9.13 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.h File Reference . . . 360
9.14 RenderAreaManager.h . . . 361
9.15 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.cpp File Reference . . . 362
9.16 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.h File Reference . . . 362
9.16.1 Detailed Description . . . 363
9.17 ShapesManager.h . . . 363
9.18 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.cpp File Reference . . . 364
9.19 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.h File Reference . . . 364
9.20 Testimonial.h . . . 365
9.21 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.cpp File Reference . . . 366
9.22 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.h File Reference . . . 366
9.23 TestimonialManager.h . . . 367
9.24 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.cpp File Reference . . . 368
9.25 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.h File Reference . . . 368
9.26 UserAccount.h . . . 369
9.27 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.cpp File Reference . . . 370
9.28 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.h File Reference . . . 370
9.29 UserManager.h . . . 371
9.30 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/ColumnEditDelegate.h File Reference . . . 372
9.31 ColumnEditDelegate.h . . . 373
9.32 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/darkstyle.qss File Reference . . . 374
9.33 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Geoo.qss File Reference . . . 374

9.34 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/lightstyle.qss File Reference . . . 374
9.35 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.cpp File Reference . . . 374
9.36 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.h File Reference . . . 374
9.37 loginwindow.h . . . 375
9.38 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/main.cpp File Reference . . . 376
9.38.1 Function Documentation . . . 377
9.38.1.1 GetConnectedClient() . . . 377
9.38.1.2 GetRenderAreaTestString() . . . 378
9.38.1.3 GetShapeTestString() . . . 378
9.38.1.4 GetUsersTestString() . . . 378
9.38.1.5 main() . . . 378
9.38.1.6 OnBadGetResponseTest() . . . 379
9.38.1.7 OnBadPostResponseTest() . . . 379
9.38.1.8 OnGoodGetResponseTest() . . . 379
9.38.1.9 OnGoodPostResponseTest() . . . 380
9.38.2 Variable Documentation . . . 380
9.38.2.1 pApp . . . 380
9.38.2.2 parse . . . 380
9.38.2.3 pClient . . . 380
9.39 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/main.cpp File Reference . . . 380
9.39.1 Detailed Description . . . 381
9.39.2 Function Documentation . . . 381
9.39.2.1 main() . . . 381
9.40 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/webservice-dockerized/main.cpp File Reference . . . 382
9.40.1 Function Documentation . . . 383
9.40.1.1 get_json_file() . . . 383
9.40.1.2 main() . . . 384
9.41 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.cpp File Reference . . . 385
9.42 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.h File Reference . . . 385
9.43 mainwindow.h . . . 386
9.44 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.ui File Reference . . . 388
9.45 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Medize.qss File Reference . . . 388
9.46 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.cpp File Reference . . . 388

9.47 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.h File Reference . . . 389
9.47.1 Detailed Description . . . 390
9.48 renderarea.h . . . 390
9.49 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/resources.qrc File Reference . . . 390
9.50 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Testimonial↩ Dialog.cpp File Reference . . . 390
9.51 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Testimonial↩ Dialog.h File Reference . . . 391
9.51.1 Detailed Description . . . 392
9.52 TestimonialDialog.h . . . 392
9.53 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Testimonials↩ DisplayDialog.cpp File Reference . . . 392
9.54 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Testimonials↩ DisplayDialog.h File Reference . . . 393
9.54.1 Detailed Description . . . 394
9.55 TestimonialsDisplayDialog.h . . . 394
9.56 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/all_shapes.h File Reference . . . 394
9.56.1 Enumeration Type Documentation . . . 395
9.56.1.1 ShapeIDs . . . 395
9.57 all_shapes.h . . . 395
9.58 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.cpp File Reference . . . 396
9.59 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.h File Reference . . . 396
9.60 circle.h . . . 397
9.61 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.cpp File Reference . . . 397
9.62 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.h File Reference . . . 398
9.63 ellipse.h . . . 398
9.64 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.cpp File Reference . . . 399
9.65 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.h File Reference 399
9.66 line.h . . . 400
9.67 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.cpp File Reference . . . 401
9.68 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.h File Reference . . . 401
9.69 polygon.h . . . 402
9.70 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.cpp File Reference . . . 402
9.71 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.h File Reference . . . 403

9.72 polyline.h . . . 403
9.73 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.cpp File Reference . . . 404
9.74 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.h File Reference . . . 404
9.75 rectangle.h . . . 405
9.76 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.cpp File Reference . . . 406
9.76.1 Function Documentation . . . 406
9.76.1.1 operator<() . . . 406
9.76.1.2 operator==() . . . 406
9.77 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.h File Reference . . . 407
9.77.1 Variable Documentation . . . 408
9.77.1.1 PI . . . 408
9.78 shape.h . . . 408
9.79 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.cpp File Reference . . . 410
9.80 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.h File Reference . . . 410
9.81 square.h . . . 411
9.82 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.cpp File Reference . . . 411
9.83 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.h File Reference 412
9.84 text.h . . . 412
9.85 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/vector.h File Reference . . . 413
9.85.1 Macro Definition Documentation . . . 414
9.85.1.1 ALPHA_VECTOR_H . . . 414
9.86 vector.h . . . 414
9.87 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/webservice/webservice.cpp File Reference . . . 418
9.87.1 Detailed Description . . . 419
9.87.2 Function Documentation . . . 420
9.87.2.1 get_json_file() . . . 420
9.87.2.2 main() . . . 421

Index **423**

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::MorphicShape 132
Parser 135
QDialog
- LoginWindow 75
- TestimonialDialog 283
- TestimonialsDisplayDialog 302

QMainWindow
- MainWindow 85

QObject
- ApiClient 15
- AppDriver 30
- RenderAreaManager 207
- ShapesManager 256
- TestimonialManager 288
- UserManager 327

QStyledItemDelegate
- ColumnEditDelegate 51

QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t 186
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t 187
QWidget
- RenderArea 199

Parser::RawUser 187
Shape 223
- Circle 42
- Ellipse 55
- Line 64
- Polygon 166
- Polyline 177
- Rectangle 189
- Square 268
- Text 305

Testimonial 277
UserAccount 320
alpha::vector< T > 341

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
Orchestrates the main application logic and connections 30
Circle
The Circle class 42
ColumnEditDelegate
The ColumnEditDelegate class 51
Ellipse
The Ellipse class 55
Line
The Line class 64
LoginWindow
A dialog window for user login and signup 75
MainWindow
The main window of the 2D Graphics Modeler application 85
Parser::MorphicShape
Internal accumulator structure for parsing shape data from JSON 132
Parser
Provides functionality for parsing JSON data to C++ objects and vice-versa 135
Polygon
The Polygon class 166
Polyline
The Polyline class 177
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t 186
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t 187
Parser::RawUser
Internal accumulator structure for parsing user account data from JSON 187
Rectangle
The Rectangle class 189
RenderArea
Canvas for drawing 2D shapes 199
RenderAreaManager
Manages the shapes to be rendered and interacts with the backend API 207
Shape
The Shape Abstract Base Class 223

ShapesManager 256
Square
The Square class 268
Testimonial
Represents a user testimonial 277
TestimonialDialog
User interface for submitting testimonials 283
TestimonialManager
Manages user testimonials, including storage, retrieval, and prompting logic 288
TestimonialsDisplayDialog
UI to display customer testimonials 302
Text
The Text class 305
UserAccount
Represents a user account within the application 320
UserManager
Manages user accounts, including creation, authentication, and persistence 327
alpha::vector< T > 341

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.cpp 351
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.h
Implements the ApiClient class that makes API requests to the Crow Webservice 351
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.cpp 353
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.h 353
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/main.cpp 376
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_ApiClient.cpp . . 355
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_UserManager.cpp
356
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.cpp 357
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h 357
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.cpp
360
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.h
360
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.cpp . 362
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.h . . 362
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.cpp . . . 364
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.h 364
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.cpp
366
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.h 366
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.cpp . . 368
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.h 368
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.cpp . . 370
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.h . . . 370
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/ColumnEditDelegate.h 372
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/darkstyle.qss 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Geoo.qss 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/lightstyle.qss 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.cpp . . . 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.h 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/main.cpp
Main entry point for the 2D Graphics Modeler application 380
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.cpp . . . 385

/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.h 385
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.ui 388
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Medize.qss 388
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.cpp . . . 388
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.h
Defines the RenderArea class, a custom QWidget for drawing and interacting with 2D shapes . 389
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/resources.qrc 390
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.cpp 390
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.h
Defines the TestimonialDialog class for submitting user testimonials 391
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialsDisplayDialog.cpp 392
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialsDisplayDialog.h
Defines the TestimonialsDisplayDialog class for showing customer testimonials 393
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/all_shapes.h 394
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.cpp 396
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.h 396
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.cpp 397
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.h 398
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.cpp 399
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.h 399
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.cpp 401
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.h 401
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.cpp 402
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.h 403
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.cpp 404
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.h 404
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.cpp 406
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.h 407
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.cpp 410
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.h 410
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.cpp 411
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.h 412
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/vector.h 413
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/webservice/webservice.cpp
Implements the Crow web service for handling shapes, render area, user, and testimonial data 418
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/webservice-dockerized/main.cpp . 382

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

API Reply Signals

Signals emitted by ApiClient to communicate request outcomes.

These signals notify listeners about the success or failure of GET, POST, and DELETE operations.

- 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

Constructor and Core API Functions

Public methods for ApiClient.

This section includes the constructor and functions for making API requests.

- ApiClient (QObject *parent=nullptr)

  Default Constructor.

Shape Data Management

API endpoints for managing Shape data.

These functions interact with the `/shapes` and `/shapes-all` endpoints to retrieve, submit, or delete shape information.

Note

These endpoints are deprecated

- void GetShapes ()

  Makes a request to get data from the Get /shapes API Endpoint.
- void PostShapes (std::string json)

  Makes a request to send data to the Post /shapes API Endpoint.
- void DeleteShapesAll ()

  Makes a request to delete all shapes via the Delete /shapes-all endpoint.

Render Area Data Management

API endpoints for managing Render Area data.

These functions interact with the `/render_area` and `/render_area-all` endpoints to retrieve, submit, or delete render area configurations.

- void GetRenderArea ()

  Makes a request to get data from the Get /render_area API Endpoint.
- void PostRenderArea (std::string json)

  Makes a request to send data to the Post /render_area API Endpoint.
- void DeleteRenderAreaAll ()

  Makes a request to delete all render area data via Delete /render_area-all.

User Data Management

API endpoints for managing User data.

These functions interact with the `/users` and `/users-all` endpoints to retrieve, submit, or delete user information.

- void GetUsers ()

Makes a request to get data from the Get /users API Endpoint.

- void PostUsers (std::string json)

  Makes a request to send data to the Post /users API Endpoint.
- void DeleteUsersAll ()

  Makes a request to delete all user data via Delete /users-all.

Testimonial Data Management

API endpoints for managing Testimonial data.

These functions interact with the `/testimonials` endpoint to retrieve, submit, or delete testimonial entries.

- void GetTestimonials ()

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

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

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

Private Slots

Network Reply Analysis

These slots are connected to the finished signal of QNetworkReply and determine whether the operation was successful, emitting the appropriate Good/Bad Get/Post/Delete reply signals.

- void AnalyzeGetReply ()

  Analyzes the reply for GET requests.
- void AnalyzePostReply ()

  Analyzes the reply for POST requests.
- void AnalyzeDeleteReply ()

  Analyzes the reply for DELETE requests.

Private Attributes

- QNetworkAccessManager * manager

  Manages all network operations for the client.

8.1.1 Detailed Description

Documentation Qt Version 6.9.0.

Manages network communication with the backend API.

This class provides methods to perform GET, POST, and DELETE requests to various endpoints of the webservice. It uses QNetworkAccessManager for handling network operations and emits signals to indicate the success or failure of these requests.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 ApiClient()

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

Default Constructor.

Here is the call graph for this function:

8.1.3 Member Function Documentation

8.1.3.1 AnalyzeDeleteReply

```
void ApiClient::AnalyzeDeleteReply ()  [private], [slot]
```

Analyzes the reply for DELETE requests.

Here is the call graph for this function:

Here is the caller graph for this function:

8.1.3.2 AnalyzeGetReply

```
void ApiClient::AnalyzeGetReply () [private], [slot]
```

Analyzes the reply for GET requests.

Here is the call graph for this function:

Here is the caller graph for this function:

8.1.3.3 AnalyzePostReply

```
void ApiClient::AnalyzePostReply () [private], [slot]
```

Analyzes the reply for POST requests.

Here is the call graph for this function:

Here is the caller graph for this function:

8.1.3.4 BadDeleteReply

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

Signal for a failed request sent to a Delete endpoint.

Here is the caller graph for this function:

8.1.3.5 BadGetReply

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

Signal for a failed request sent to a Get endpoint.

Here is the caller graph for this function:

8.1.3.6 BadPostReply

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

Signal for a failed request sent to a Post endpoint.

Here is the caller graph for this function:

8.1.3.7 DeleteRenderAreaAll()

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

Makes a request to delete all render area data via Delete /render_area-all.

Here is the call graph for this function:

8.1.3.8 DeleteShapesAll()

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

Makes a request to delete all shapes via the Delete /shapes-all endpoint.

Here is the call graph for this function:

8.1.3.9 DeleteTestimonialsAll()

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

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

Here is the call graph for this function:

8.1.3.10 DeleteUsersAll()

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

Makes a request to delete all user data via Delete /users-all.

Here is the call graph for this function:

8.1.3.11 GetRenderArea()

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

Makes a request to get data from the Get /render_area API Endpoint.

Here is the call graph for this function:

8.1.3.12 GetShapes()

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

Makes a request to get data from the Get /shapes API Endpoint.

Here is the call graph for this function:

8.1.3.13 GetTestimonials()

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

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

Here is the call graph for this function:

8.1.3.14 GetUsers()

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

Makes a request to get data from the Get /users API Endpoint.

Here is the call graph for this function:

8.1.3.15 GoodDeleteReply

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

Signal for a successful request sent to a Delete endpoint.

Here is the caller graph for this function:

8.1.3.16 GoodGetReply

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

Signal for a successful request sent to a Get endpoint.

Here is the caller graph for this function:

8.1.3.17 GoodPostReply

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

Signal for a successful request sent to a Post endpoint.

Here is the caller graph for this function:

8.1.3.18 PostRenderArea()

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

Makes a request to send data to the Post /render_area API Endpoint.

Here is the call graph for this function:

8.1.3.19 PostShapes()

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

Makes a request to send data to the Post /shapes API Endpoint.

Here is the call graph for this function:

8.1.3.20 PostTestimonials()

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

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

Here is the call graph for this function:

8.1.3.21 PostUsers()

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

Makes a request to send data to the Post /users API Endpoint.

Here is the call graph for this function:

8.1.4 Member Data Documentation

8.1.4.1 manager

```
QNetworkAccessManager* ApiClient::manager  [private]
```

Manages all network operations for the client.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.cpp
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_ApiClient.cpp

8.2 AppDriver Class Reference

Orchestrates the main application logic and connections.

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

Inheritance diagram for AppDriver:

Collaboration diagram for AppDriver:

Public Member Functions

Core Application Lifecycle

Public interface for the AppDriver.

This section includes the constructor, destructor, and core methods for running, shutting down, and loading data for the application.

- AppDriver (QObject ∗parent=nullptr)

  Constructs the AppDriver and initializes manager objects.
- ∼AppDriver ()

  Destroys the AppDriver and cleans up manager objects.
- void run ()

  Initializes and shows the main application window and loads initial data.
- void shutdown ()

  Saves data from managers before the application closes.
- void loadAllData ()

  Triggers the loading of all necessary data by the managers.

Private Slots

Render Area UI Slots

Slots connecting the render area UI to RenderAreaManager data.

These slots handle signals from the frontend render area, such as shape additions, modifications, and deletions, and delegate these actions to the RenderAreaManager.

- void onRenderShapeAdded (Shape ∗shape)

  Handles the addition of a new shape from the render area.
- void onRenderShapeChanged (Shape ∗shape, QString key, int value)
- void onRenderShapeDeleted (const int trackerId)

  Handles the deletion of a single shape from the render area.
- void onRenderDeleteAllShapes ()

  Handles the request to delete all shapes from the render area.

User Management UI Slots

Slots connecting the user login UI to UserManager data.

These slots handle signals related to user management from the frontend, such as new user creation, credential modification, user deletion, and login attempts, delegating these actions to the UserManager.

- void onNewUser (const QString username, const QString password, const bool admin)
- void onUserModified (const QString username, const QString password, const bool admin)
- void onUserDeleted (const QString username)

  Handles the deletion of a single user.
- void onDeleteAllUsers ()

  Handles the request to delete all users.
- void onLoginAttempt (const QString username, const QString password)

Private Member Functions

Connection Setup Helpers

Private helper methods for setting up connections.

These subroutines are called during initialization to establish signal-slot connections between different components of the application.

- void connectFrontendToDriver ()

  Connects signals from the frontend UI to slots in this AppDriver.
- void connectManagersToFrontend ()

  Connects signals from backend managers to slots in the frontend UI.

Private Attributes

Core Components

Private member variables for the AppDriver.

These members hold pointers to the core manager classes and the main window.

- MainWindow * mainWindow
  Pointer to the main application window.
- RenderAreaManager * renderedShapes
  Manages all the shapes that are currently rendered.
- UserManager * user
  Manages the current user and holds all existing users to authenticate against.

8.2.1 Detailed Description

Orchestrates the main application logic and connections.

The AppDriver class is responsible for initializing and managing the main components of the application, including the RenderAreaManager, UserManager, and the MainWindow. It handles the setup of signal-slot connections between the frontend UI elements and the backend data managers. It also manages the application lifecycle, including startup, data loading, and shutdown procedures.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 AppDriver()

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

Constructs the AppDriver and initializes manager objects.

8.2.2.2 ~AppDriver()

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

Destroys the AppDriver and cleans up manager objects.

8.2.3 Member Function Documentation

8.2.3.1 connectFrontendToDriver()

```
void AppDriver::connectFrontendToDriver () [private]
```

Connects signals from the frontend UI to slots in this AppDriver.

Here is the call graph for this function:

Here is the caller graph for this function:

8.2.3.2 connectManagersToFrontend()

```
void AppDriver::connectManagersToFrontend () [private]
```

Connects signals from backend managers to slots in the frontend UI.

Here is the call graph for this function:

Here is the caller graph for this function:

8.2.3.3 loadAllData()

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

Triggers the loading of all necessary data by the managers.

Here is the caller graph for this function:

8.2.3.4 onDeleteAllUsers

```
void AppDriver::onDeleteAllUsers () [private], [slot]
```

Handles the request to delete all users.

8.2.3.5 onLoginAttempt

```
void AppDriver::onLoginAttempt (
            const QString username,
            const QString password) [private], [slot]
```

Parameters

username	Handles a user login attempt.

Here is the caller graph for this function:

8.2.3.6 onNewUser

```
void AppDriver::onNewUser (
            const QString username,
            const QString password,
            const bool admin)  [private], [slot]
```

Parameters

username	Handles the creation of a new user.

Here is the caller graph for this function:

8.2.3.7 onRenderDeleteAllShapes

```
void AppDriver::onRenderDeleteAllShapes ()  [private], [slot]
```

Handles the request to delete all shapes from the render area.

Here is the caller graph for this function:

8.2.3.8 onRenderShapeAdded

```
void AppDriver::onRenderShapeAdded (
            Shape * shape)  [private], [slot]
```

Handles the addition of a new shape from the render area.

Here is the caller graph for this function:

8.2.3.9 onRenderShapeChanged

```
void AppDriver::onRenderShapeChanged (
            Shape * shape,
            QString key,
            int value)  [private], [slot]
```

Parameters

shape	Handles modifications to an existing shape from the render area.

Here is the caller graph for this function:

8.2.3.10 onRenderShapeDeleted

```
void AppDriver::onRenderShapeDeleted (
            const int trackerId)  [private], [slot]
```

Handles the deletion of a single shape from the render area.

Here is the caller graph for this function:

8.2.3.11 onUserDeleted

```
void AppDriver::onUserDeleted (
            const QString username)  [private], [slot]
```

Handles the deletion of a single user.

8.2.3.12 onUserModified

```
void AppDriver::onUserModified (
            const QString username,
            const QString password,
            const bool admin)  [private], [slot]
```

Parameters

username	Handles modification of an existing user's credentials.

8.2.3.13 run()

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

Initializes and shows the main application window and loads initial data.

Here is the call graph for this function:

Here is the caller graph for this function:

8.2.3.14 shutdown()

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

Saves data from managers before the application closes.

8.2.4 Member Data Documentation

8.2.4.1 mainWindow

```
MainWindow* AppDriver::mainWindow [private]
```

Pointer to the main application window.

8.2.4.2 renderedShapes

```
RenderAreaManager* AppDriver::renderedShapes [private]
```

Manages all the shapes that are currently rendered.

8.2.4.3 user

```
UserManager* AppDriver::user [private]
```

Manages the current user and holds all existing users to authenticate against.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/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 (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- int r

  Mutator Functions.

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

Here is the caller graph for this function:

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:

8.3.4 Member Data Documentation

8.3.4.1 r

```
int Circle::r [private]
```

Mutator Functions.

int radius of the circle

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.cpp

8.4 ColumnEditDelegate Class Reference

The ColumnEditDelegate class.

```
#include <ColumnEditDelegate.h>
```

Inheritance diagram for ColumnEditDelegate:

Collaboration diagram for ColumnEditDelegate:

Public Member Functions

- ColumnEditDelegate (QObject ∗parent=nullptr)

  ColumnEditDelegate Constructor.
- QWidget ∗ createEditor (QWidget ∗parent, const QStyleOptionViewItem &option, const QModelIndex &index) const override

  createEditor
- void setCanEdit (bool edit)

  setCanEdit

Private Attributes

- bool canEdit = false

  Flag to control whether editing is allowed.

8.4.1 Detailed Description

The ColumnEditDelegate class.

8.4.2 Constructor & Destructor Documentation

8.4.2.1 ColumnEditDelegate()

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

ColumnEditDelegate Constructor.

Parameters

parent	

8.4.3 Member Function Documentation

8.4.3.1 createEditor()

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

createEditor

Parameters

parent	
option	
index	

Returns

8.4.3.2 setCanEdit()

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

setCanEdit

Parameters

edit	

8.4.4 Member Data Documentation

8.4.4.1 canEdit

```
bool ColumnEditDelegate::canEdit = false  [private]
```

Flag to control whether editing is allowed.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/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 (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- int a

  Mutator Functions.
- int b

  Semi-Major Axis.

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.

Here is the caller graph for this function:

8.5.3.4 getB()

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

Here is the caller graph for this function:

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:

8.5.4 Member Data Documentation

8.5.4.1 a

```
int Ellipse::a [private]
```

Mutator Functions.

Semi-Minor Axis

8.5.4.2 b

```
int Ellipse::b [private]
```

Semi-Major Axis.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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)

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

Public Member Functions inherited from Shape

- Shape (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- QPoint startPoint

  Starting point of the Line.
- QPoint endPoint

  Ending point of the line.

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)
```

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:

8.6.4 Member Data Documentation

8.6.4.1 endPoint

```
QPoint Line::endPoint  [private]
```

Ending point of the line.

8.6.4.2 startPoint

```
QPoint Line::startPoint  [private]
```

Starting point of the Line.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.cpp

8.7 LoginWindow Class Reference

A dialog window for user login and signup.

```
#include <loginwindow.h>
```

Inheritance diagram for LoginWindow:

Collaboration diagram for LoginWindow:

Signals

User Action Signals

Signals emitted based on user interactions within the dialog.

These signals notify other parts of the application about login or signup attempts.

- void loginRequested (const QString &username, const QString &password)

Emitted when the user attempts to log in with the provided credentials.

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

  Emitted when the user attempts to sign up with the provided credentials. (Admin flag is currently hardcoded to true in the .cpp).

Public Member Functions

Constructor and Public Accessors

Methods for creating and interacting with the LoginWindow.

Includes the constructor and methods to retrieve user input.

- LoginWindow (QWidget *parent=nullptr)

  Constructor that sets up the UI components for login and signup.
- QString username () const

  Returns the username entered in the currently active input field (login or signup).
- QString password () const

  Returns the password entered in the currently active input field (login or signup).

Private Slots

Page Navigation and Action Slots

Slots to handle UI events like button clicks.

These slots manage switching between login/signup pages and triggering login/signup attempts.

- void showSignupPage ()

  Switches the QStackedWidget to display the signup page.
- void showLoginPage ()

  Switches the QStackedWidget to display the login page.
- void attemptLogin ()

  Emits the loginRequested signal with current input from login fields.
- void attemptSignup ()

  Emits the signupRequested signal with current input from signup fields.

Private Attributes

UI Components

Private member variables storing pointers to UI widgets.

These widgets make up the login and signup pages.

- QStackedWidget * stack

  The stacked widget managing the login and signup pages.
- QWidget * loginPage

  The widget container for the login page UI elements.
- QLineEdit * loginUserEdit

  Input field for the username on the login page.
- QLineEdit * loginPassEdit

  Input field for the password on the login page.
- QPushButton * loginBtn

  Button to initiate a login attempt.
- QPushButton * toSignupBtn

  Button to navigate to the signup page.
- QWidget * signupPage

  The widget container for the signup page UI elements.
- QLineEdit * signupUserEdit

  Input field for the username on the signup page.
- QLineEdit * signupPassEdit

  Input field for the password on the signup page.
- QPushButton * signupBtn

  Button to initiate a signup attempt.
- QPushButton * backToLoginBtn

  Button to navigate back to the login page.

8.7.1 Detailed Description

A dialog window for user login and signup.

This class provides a QDialog with a QStackedWidget to switch between a login page and a signup page. It emits signals when a login or signup attempt is made.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 LoginWindow()

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

Constructor that sets up the UI components for login and signup.

Here is the call graph for this function:

8.7.3 Member Function Documentation

8.7.3.1 attemptLogin

```
void LoginWindow::attemptLogin ()  [private], [slot]
```

Emits the loginRequested signal with current input from login fields.

Here is the call graph for this function:

Here is the caller graph for this function:

8.7.3.2 attemptSignup

```
void LoginWindow::attemptSignup () [private], [slot]
```

Emits the signupRequested signal with current input from signup fields.

Here is the call graph for this function:

Here is the caller graph for this function:

8.7.3.3 loginRequested

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

Emitted when the user attempts to log in with the provided credentials.

Here is the call graph for this function:

Here is the caller graph for this function:

8.7.3.4 password()

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

Returns the password entered in the currently active input field (login or signup).

Here is the caller graph for this function:

8.7.3.5 showLoginPage

```
void LoginWindow::showLoginPage () [private], [slot]
```

Switches the QStackedWidget to display the login page.

Here is the caller graph for this function:

8.7.3.6 showSignupPage

```
void LoginWindow::showSignupPage () [private], [slot]
```

Switches the QStackedWidget to display the signup page.

Here is the caller graph for this function:

8.7.3.7 signupRequested

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

Emitted when the user attempts to sign up with the provided credentials. (Admin flag is currently hardcoded to true in the .cpp).

Here is the call graph for this function:

Here is the caller graph for this function:

8.7.3.8 username()

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

Returns the username entered in the currently active input field (login or signup).

Here is the caller graph for this function:

8.7.4 Member Data Documentation

8.7.4.1 backToLoginBtn

```
QPushButton* LoginWindow::backToLoginBtn [private]
```

Button to navigate back to the login page.

8.7.4.2 loginBtn

```
QPushButton* LoginWindow::loginBtn [private]
```

Button to initiate a login attempt.

8.7.4.3 loginPage

```
QWidget* LoginWindow::loginPage [private]
```

The widget container for the login page UI elements.

8.7.4.4 loginPassEdit

```
QLineEdit* LoginWindow::loginPassEdit [private]
```

Input field for the password on the login page.

8.7.4.5 loginUserEdit

```
QLineEdit* LoginWindow::loginUserEdit [private]
```

Input field for the username on the login page.

8.7.4.6 signupBtn

```
QPushButton* LoginWindow::signupBtn [private]
```

Button to initiate a signup attempt.

8.7.4.7 signupPage

```
QWidget* LoginWindow::signupPage [private]
```

The widget container for the signup page UI elements.

8.7.4.8 signupPassEdit

```
QLineEdit* LoginWindow::signupPassEdit [private]
```

Input field for the password on the signup page.

8.7.4.9 signupUserEdit

```
QLineEdit* LoginWindow::signupUserEdit [private]
```

Input field for the username on the signup page.

8.7.4.10 stack

```
QStackedWidget* LoginWindow::stack [private]
```

The stacked widget managing the login and signup pages.

8.7.4.11 toSignupBtn

```
QPushButton* LoginWindow::toSignupBtn [private]
```

Button to navigate to the signup page.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.cpp

8.8 MainWindow Class Reference

The main window of the 2D Graphics Modeler application.

```
#include <mainwindow.h>
```

Inheritance diagram for MainWindow:

Collaboration diagram for MainWindow:

Public Slots

RenderAreaManager Response Slots

Slots for handling responses from RenderAreaManager.

These slots are connected to signals from the RenderAreaManager to update the UI when the state of rendered shapes changes or status messages are available.

- void onRenderAreaChanged ()
  Slot triggered when the render area's content has changed, requiring a UI update.
- void onRenderAreaNotChanged (const QString &message)
  Slot triggered when an operation on the render area did not result in a change (e.g., error).
- void showRenderStatusMessage (const QString &message)
  Displays a status message related to rendering operations.

Login Process Management Slots

Slots for managing the login process.

These slots handle user interactions related to logging in and signing up.

- void onLoginClicked ()
  Slot triggered when the login button is clicked, typically opening a login dialog.
- void onLoginRequest (const QString &username, const QString &password)
  Slot triggered by a login dialog to initiate a login attempt.
- void onSignupRequest (const QString &username, const QString &password, const bool admin)

UserManager Response Slots

Slots for handling responses from UserManager.

These slots are connected to signals from the UserManager to update the UI based on the outcome of user authentication attempts.

- void onUserAuthentication (const UserAccount *currUser)
  Slot triggered upon successful user authentication. Updates UI with user info.
- void onUserAuthenticationFailure (const QString &message)
  Slot triggered when user authentication fails. Displays an error message.

Signals

Shape Manipulation Signals

Signals related to shape manipulation.

These signals are emitted to notify other components (likely AppDriver or RenderAreaManager) about actions performed on shapes by the user through the UI.

- void shapeAdded (Shape *shape)
  Emitted when a new shape is added by the user.
- void shapeChanged (Shape *shape, QString key, int value)
  Emitted when a property of an existing shape is changed.
- void displayedTextChanged (Text *text, QString newText)
  Emitted when the displayed text of a Text shape is changed.
- void shapeDeleted (int trackerId)
  Emitted when a shape is requested to be deleted.
- void deleteAllShapes ()
  Emitted when a request to delete all shapes is made.

User Authentication Signals

Signals related to user login and signup.

These signals are emitted to the AppDriver or UserManager to handle user authentication and account creation processes.

- void loginAttempt (const QString &username, const QString &password)
  Emitted when a user attempts to log in.
- void newUserAdded (const QString &username, const QString &password, const bool admin)

Dialog Flow Control Signals

Signals for controlling dialog flow.

These signals are used to communicate the success or failure of operations like login, often to control the state of dialog windows.

- void loginSuccess ()
  Emitted when a login attempt is successful.
- void loginFailed (const QString &message)
  Emitted when a login attempt fails, providing an error message.

Public Member Functions

Constructors and Destructor

Manages the lifecycle of MainWindow objects.

These methods handle the creation, initialization, and cleanup of the MainWindow object.

- MainWindow (QWidget *parent=nullptr)
  Default constructor.
- MainWindow (QWidget *parent, const alpha::vector< Shape * > *renderedShapes, const UserAccount *currUser)
- ~MainWindow ()
  Destructor. Cleans up UI elements and other resources.

Shape Display Interface

Public interface methods for shape display.

These methods are responsible for drawing shapes on the render area and populating the tree widget with shape information.

- void drawShapes () const
  Instructs the render area to draw the current shapes. (Note: Actual drawing is typically triggered by `render↩Area->update()`*).*
- void shapes_to_treeWidget ()
  Populates the tree widget with details of the current shapes and sets up editing controls.

Private Slots

UI Customization Slots

Slot for UI style toggling.

This slot handles changes to the application's visual style.

- void onToggleStyle (bool checked=true)
  Toggles the application's stylesheet between predefined styles.

Shape Creation Action Slots

Slots for shape creation actions.

These slots are connected to toolbar buttons or menu actions that allow the user to add new shapes to the canvas.

- void on_actionnew_line_button_triggered ()
  Slot triggered to create and add a new Line shape.
- void on_actionnew_square_button_triggered ()
  Slot triggered to create and add a new Square shape.
- void on_actionnew_rectange_button_triggered ()

Slot triggered to create and add a new Rectangle shape.

- void on_actionnew_circle_button_triggered ()

  Slot triggered to create and add a new Circle shape.
- void on_actionnew_ellipse_button_triggered ()

  Slot triggered to create and add a new Ellipse shape.
- void on_actionnew_polyline_button_triggered ()

  Slot triggered to create and add a new Polyline shape.
- void on_actionnew_polygon_button_triggered ()

  Slot triggered to create and add a new Polygon shape.
- void on_actionnew_text_button_triggered ()

  Slot triggered to create and add a new Text shape.
- void on_actionremove_shape_button_triggered ()

  Slot triggered to remove the currently selected shape.

Shape Report Sorting Slots

Slots for shape sorting functionality in the shape report.

This slot handles changes in the sorting criteria for the shape table.

- void onSortMethodChanged (int index)

  Slot triggered when the sort method or order in the shape report table changes.

Tree Widget Editing Slots

Slots for modifying shape properties via the tree widget.

These slots respond to user edits made in the shape properties tree, updating the corresponding shape attributes.

- void onTreeWidgetItemChanged (QTreeWidgetItem *item, int column)

  Slot triggered when an item in the shape properties tree widget is changed.
- void onComboBoxChanged (int newIndex)

  Slot triggered when a QComboBox value in the tree widget changes.
- void onSpinBoxChanged ()

  Slot triggered when a QSpinBox value in the tree widget changes.

Testimonial UI Slots

Slots for testimonial UI.

These slots handle the display of testimonial prompts and the testimonial viewing dialog.

- void showTestimonialPrompt ()

  Displays a dialog prompting the user to submit a testimonial.
- void showTestimonialsDisplay ()

  Displays a dialog showing existing testimonials.

Contact Information Slot

Slot for displaying the "Contact Us" window.

This slot is triggered by a menu action to show contact information.

- void onContactUsClicked ()

  Slot triggered to display the 'Contact Us' information window.

Private Member Functions

UI Management Helpers

Private helper methods for UI management.

These methods assist in managing UI elements, such as adding shapes to the tree widget and loading stylesheets.

- void addToShapeTree (Shape *shape)
  Adds a given shape to the properties tree widget and emits shapeAdded signal.
- QString loadStyleSheet (const QString &path)
  Loads a Qt stylesheet from the specified file path.

Editor Widget Factories

Factory methods for creating QComboBox and QSpinBox widgets.

These methods are used to dynamically create and configure editor widgets (like QComboBoxes for color, style, etc., and QSpinBoxes for numerical values) for the shape properties tree. Each method initializes the widget with items and sets its current value based on the shape's property.

- QComboBox * createShapeTypeComboBox (const QString &currentShapeType)
  Creates and configures a QComboBox for selecting shape types.
- QSpinBox * createPenWidthSpinBox (int currentPenWidth)
  Creates and configures a QSpinBox for editing pen width.
- QComboBox * createColorComboBox (int currentColor)
  Creates and configures a QComboBox for selecting colors.
- QComboBox * createPenStyleComboBox (int currentPenStyle)
  Creates and configures a QComboBox for selecting pen styles.
- QComboBox * createPenCapStyleComboBox (int currentPenCapStyle)
  Creates and configures a QComboBox for selecting pen cap styles.
- QComboBox * createPenJoinStyleComboBox (int currentPenJoinStyle)
  Creates and configures a QComboBox for selecting pen join styles.
- QComboBox * createBrushStyleComboBox (int currentBrushStyle)
  Creates and configures a QComboBox for selecting brush styles.
- QComboBox * createAlignmentComboBox (Qt::AlignmentFlag currentAlignment)
  Creates and configures a QComboBox for selecting text alignment.
- QComboBox * createFontComboBox (QFont currentFont)
  Creates and configures a QComboBox for selecting font families.
- QComboBox * createFontStyleComboBox (int currentFontStyle)
  Creates and configures a QComboBox for selecting font styles (normal, italic, oblique).
- QComboBox * createFontWeightComboBox (QFont::Weight currentFontWeight)
  Creates and configures a QComboBox for selecting font weights.

Testimonial Feature Setup

Testimonial feature setup.

This method initializes components related to the testimonial feature.

- void setupTestimonials ()
  Sets up connections and UI elements for the testimonial feature.

Private Attributes

Core UI Components and Data

Core UI components and data pointers.

These members are essential for the MainWindow's operation, including the UI definition, rendering area, and pointers to shared data.

- Ui::MainWindow ∗ ui

  Pointer to the auto-generated UI class.
- RenderArea ∗ renderArea

  Pointer to the custom widget responsible for drawing shapes.
- const alpha::vector< Shape ∗ > ∗ renderShapes

  Pointer to the vector of shapes currently being rendered.
- const UserAccount ∗ currUser

  Pointer to the currently logged-in user account.
- QLabel ∗ userStatusLabel

  Label in the status bar to display the current user's login status.
- ColumnEditDelegate ∗ delegate

  Custom delegate for editing items in the shape properties tree widget.

Contact Us Window Members

Members for the "Contact Us" window.

Pointers to widgets used in the "Contact Us" information display.

- QWidget ∗ contactUsWidget

  Pointer to the main widget for the Contact Us display.
- QLabel ∗ logoLabel

  Label to display the team logo in the Contact Us display.
- QLabel ∗ teamNameLabel

  Label to display the team name in the Contact Us display.
- QWidget ∗ contactWindow

  Pointer to the QWidget that serves as the 'Contact Us' window.

Shape Report Table Management

Methods and members related to the shape report table.

These are used to create, populate, and manage the tab/window that displays a sortable table of shape properties.

- QTabWidget ∗ tabWidget

  Pointer to the tab widget (if used for the shape report, otherwise could be for other purposes).
- QTableWidget ∗ shapeTable

  Pointer to the table widget used to display shape properties in the report.
- QComboBox ∗ sortDropdown

  Dropdown QComboBox for selecting the sorting criterion in the shape report.
- QComboBox ∗ sortOrderDropdown

  Dropdown QComboBox for selecting the sorting order (ascending/descending) in the shape report.
- void createShapeTableTab ()

  Creates and displays a new tab or window containing the shape report table.

Shape Sorting Logic

Sorting algorithm and comparison functions for the shape report.

Implements a selection sort algorithm and provides static comparison functions used to sort shapes by different criteria.

- void selection_sort (alpha::vector< Shape * > &shapes, bool(*compare)(const Shape *, const Shape *), bool ascending)
- void populateShapeTable (const alpha::vector< Shape * > &shapes)

  Populates the shape report table with data from the provided vector of shapes.
- static bool sortById (const Shape *a, const Shape *b)

  Static comparison function to sort shapes by their ID.
- static bool sortByArea (const Shape *a, const Shape *b)

  Static comparison function to sort shapes by their calculated area.
- static bool sortByPerimeter (const Shape *a, const Shape *b)

  Static comparison function to sort shapes by their calculated perimeter.

8.8.1 Detailed Description

The main window of the 2D Graphics Modeler application.

This class serves as the primary user interface, managing shape rendering, user interactions, login, testimonials, and displaying shape information. It coordinates between the UI elements and backend logic.

8.8.2 Constructor & Destructor Documentation

8.8.2.1 MainWindow() [1/2]

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

Default constructor.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.2.2 MainWindow() [2/2]

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

Parameters

parent	Constructor initializing with shapes and user data.

Here is the call graph for this function:

8.8.2.3 ~MainWindow()

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

Destructor. Cleans up UI elements and other resources.

8.8.3 Member Function Documentation

8.8.3.1 addToShapeTree()

```
void MainWindow::addToShapeTree (
            Shape * shape)  [private]
```

Adds a given shape to the properties tree widget and emits shapeAdded signal.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.2 createAlignmentComboBox()

```
QComboBox * MainWindow::createAlignmentComboBox (
            Qt::AlignmentFlag currentAlignment)  [private]
```

Creates and configures a QComboBox for selecting text alignment.

Here is the caller graph for this function:

8.8.3.3 createBrushStyleComboBox()

```
QComboBox * MainWindow::createBrushStyleComboBox (
            int currentBrushStyle)  [private]
```

Creates and configures a QComboBox for selecting brush styles.

Here is the caller graph for this function:

8.8.3.4 createColorComboBox()

```
QComboBox * MainWindow::createColorComboBox (
            int currentColor)  [private]
```

Creates and configures a QComboBox for selecting colors.

Here is the caller graph for this function:

8.8.3.5 createFontComboBox()

```
QComboBox * MainWindow::createFontComboBox (
            QFont currentFont)  [private]
```

Creates and configures a QComboBox for selecting font families.

Here is the caller graph for this function:

8.8.3.6 createFontStyleComboBox()

```
QComboBox * MainWindow::createFontStyleComboBox (
            int currentFontStyle)  [private]
```

Creates and configures a QComboBox for selecting font styles (normal, italic, oblique).

Here is the caller graph for this function:

8.8.3.7 createFontWeightComboBox()

```
QComboBox * MainWindow::createFontWeightComboBox (
            QFont::Weight currentFontWeight)  [private]
```

Creates and configures a QComboBox for selecting font weights.

Here is the caller graph for this function:

8.8.3.8 createPenCapStyleComboBox()

```
QComboBox * MainWindow::createPenCapStyleComboBox (
            int currentPenCapStyle)  [private]
```

Creates and configures a QComboBox for selecting pen cap styles.

Here is the caller graph for this function:

8.8.3.9 createPenJoinStyleComboBox()

```
QComboBox * MainWindow::createPenJoinStyleComboBox (
            int currentPenJoinStyle) [private]
```

Creates and configures a QComboBox for selecting pen join styles.

Here is the caller graph for this function:

8.8.3.10 createPenStyleComboBox()

```
QComboBox * MainWindow::createPenStyleComboBox (
            int currentPenStyle) [private]
```

Creates and configures a QComboBox for selecting pen styles.

Here is the caller graph for this function:

8.8.3.11 createPenWidthSpinBox()

```
QSpinBox * MainWindow::createPenWidthSpinBox (
            int currentPenWidth)  [private]
```

Creates and configures a QSpinBox for editing pen width.

Here is the caller graph for this function:

8.8.3.12 createShapeTableTab()

```
void MainWindow::createShapeTableTab ()  [private]
```

Creates and displays a new tab or window containing the shape report table.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.13 createShapeTypeComboBox()

```
QComboBox * MainWindow::createShapeTypeComboBox (
            const QString & currentShapeType)  [private]
```

Creates and configures a QComboBox for selecting shape types.

8.8.3.14 deleteAllShapes

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

Emitted when a request to delete all shapes is made.

Here is the caller graph for this function:

8.8.3.15 displayedTextChanged

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

Emitted when the displayed text of a Text shape is changed.

Here is the caller graph for this function:

8.8.3.16 drawShapes()

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

Instructs the render area to draw the current shapes. (Note: Actual drawing is typically triggered by `renderArea->update()`).

8.8.3.17 loadStyleSheet()

```
QString MainWindow::loadStyleSheet (
            const QString & path)  [private]
```

Loads a Qt stylesheet from the specified file path.

Here is the caller graph for this function:

8.8.3.18 loginAttempt

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

Emitted when a user attempts to log in.

Here is the caller graph for this function:

8.8.3.19 loginFailed

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

Emitted when a login attempt fails, providing an error message.

Here is the caller graph for this function:

8.8.3.20 loginSuccess

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

Emitted when a login attempt is successful.

Here is the caller graph for this function:

8.8.3.21 newUserAdded

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

Parameters

password	Emitted when a new user account is requested to be created.

Here is the caller graph for this function:

8.8.3.22 on_actionnew_circle_button_triggered

```
void MainWindow::on_actionnew_circle_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Circle shape.

Here is the call graph for this function:

8.8.3.23 on_actionnew_ellipse_button_triggered

```
void MainWindow::on_actionnew_ellipse_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Ellipse shape.

Here is the call graph for this function:

8.8.3.24 on_actionnew_line_button_triggered

```
void MainWindow::on_actionnew_line_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Line shape.

Here is the call graph for this function:

8.8.3.25 on_actionnew_polygon_button_triggered

```
void MainWindow::on_actionnew_polygon_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Polygon shape.

Here is the call graph for this function:

8.8.3.26 on_actionnew_polyline_button_triggered

```
void MainWindow::on_actionnew_polyline_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Polyline shape.

Here is the call graph for this function:

8.8.3.27 on_actionnew_rectange_button_triggered

```
void MainWindow::on_actionnew_rectange_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Rectangle shape.

Here is the call graph for this function:

8.8.3.28 on_actionnew_square_button_triggered

```
void MainWindow::on_actionnew_square_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Square shape.

Here is the call graph for this function:

8.8.3.29 on_actionnew_text_button_triggered

```
void MainWindow::on_actionnew_text_button_triggered () [private], [slot]
```

Slot triggered to create and add a new Text shape.

Here is the call graph for this function:

8.8.3.30 on_actionremove_shape_button_triggered

```
void MainWindow::on_actionremove_shape_button_triggered () [private], [slot]
```

Slot triggered to remove the currently selected shape.

Here is the call graph for this function:

8.8.3.31 onComboBoxChanged

```
void MainWindow::onComboBoxChanged (
            int newIndex)  [private], [slot]
```

Slot triggered when a QComboBox value in the tree widget changes.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.32 onContactUsClicked

```
void MainWindow::onContactUsClicked () [private], [slot]
```

Slot triggered to display the 'Contact Us' information window.

Here is the caller graph for this function:

8.8.3.33 onLoginClicked

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

Slot triggered when the login button is clicked, typically opening a login dialog.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.34 onLoginRequest

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

Slot triggered by a login dialog to initiate a login attempt.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.35 onRenderAreaChanged

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

Slot triggered when the render area's content has changed, requiring a UI update.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.36 onRenderAreaNotChanged

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

Slot triggered when an operation on the render area did not result in a change (e.g., error).

Here is the caller graph for this function:

8.8.3.37 onSignupRequest

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

Parameters

password	Slot triggered by a login/signup dialog to initiate account creation.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.38 onSortMethodChanged

```
void MainWindow::onSortMethodChanged (
            int index)  [private], [slot]
```

Slot triggered when the sort method or order in the shape report table changes.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.39 onSpinBoxChanged

```
void MainWindow::onSpinBoxChanged ()  [private], [slot]
```

Slot triggered when a QSpinBox value in the tree widget changes.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.40 onToggleStyle

```
void MainWindow::onToggleStyle (
            bool checked = true)  [private], [slot]
```

Toggles the application's stylesheet between predefined styles.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.41 onTreeWidgetItemChanged

```
void MainWindow::onTreeWidgetItemChanged (
            QTreeWidgetItem * item,
            int column)  [private], [slot]
```

Slot triggered when an item in the shape properties tree widget is changed.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.42 onUserAuthentication

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

Slot triggered upon successful user authentication. Updates UI with user info.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.43 onUserAuthenticationFailure

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

Slot triggered when user authentication fails. Displays an error message.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.44 populateShapeTable()

```
void MainWindow::populateShapeTable (
            const alpha::vector< Shape * > & shapes)  [private]
```

Populates the shape report table with data from the provided vector of shapes.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.45 selection_sort()

```
void MainWindow::selection_sort (
            alpha::vector< Shape * > & shapes,
            bool(* compare )(const Shape *, const Shape *),
            bool ascending)  [private]
```

Parameters

shapes	Implements the selection sort algorithm for a vector of shapes.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.46 setupTestimonials()

```
void MainWindow::setupTestimonials ()  [private]
```

Sets up connections and UI elements for the testimonial feature.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.47 shapeAdded

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

Emitted when a new shape is added by the user.

Here is the caller graph for this function:

8.8.3.48 shapeChanged

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

Emitted when a property of an existing shape is changed.

Here is the caller graph for this function:

8.8.3.49 shapeDeleted

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

Emitted when a shape is requested to be deleted.

Here is the caller graph for this function:

8.8.3.50 shapes_to_treeWidget()

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

Populates the tree widget with details of the current shapes and sets up editing controls.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.51 showRenderStatusMessage

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

Displays a status message related to rendering operations.

Here is the caller graph for this function:

8.8.3.52 showTestimonialPrompt

```
void MainWindow::showTestimonialPrompt ()  [private], [slot]
```

Displays a dialog prompting the user to submit a testimonial.

Here is the caller graph for this function:

8.8.3.53 showTestimonialsDisplay

```
void MainWindow::showTestimonialsDisplay () [private], [slot]
```

Displays a dialog showing existing testimonials.

Here is the caller graph for this function:

8.8.3.54 sortByArea()

```
bool MainWindow::sortByArea (
            const Shape * a,
            const Shape * b) [static], [private]
```

Static comparison function to sort shapes by their calculated area.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.55 sortById()

```
bool MainWindow::sortById (
            const Shape * a,
            const Shape * b)  [static], [private]
```

Static comparison function to sort shapes by their ID.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.3.56 sortByPerimeter()

```
bool MainWindow::sortByPerimeter (
            const Shape * a,
            const Shape * b)  [static], [private]
```

Static comparison function to sort shapes by their calculated perimeter.

Here is the call graph for this function:

Here is the caller graph for this function:

8.8.4 Member Data Documentation

8.8.4.1 contactUsWidget

```
QWidget* MainWindow::contactUsWidget [private]
```

Pointer to the main widget for the Contact Us display.

8.8.4.2 contactWindow

```
QWidget* MainWindow::contactWindow [private]
```

Pointer to the QWidget that serves as the 'Contact Us' window.

8.8.4.3 currUser

```
const UserAccount* MainWindow::currUser [private]
```

Pointer to the currently logged-in user account.

8.8.4.4 delegate

```
ColumnEditDelegate* MainWindow::delegate [private]
```

Custom delegate for editing items in the shape properties tree widget.

8.8.4.5 logoLabel

```
QLabel* MainWindow::logoLabel [private]
```

Label to display the team logo in the Contact Us display.

8.8.4.6 renderArea

```
RenderArea* MainWindow::renderArea [private]
```

Pointer to the custom widget responsible for drawing shapes.

8.8.4.7 renderShapes

```
const alpha::vector<Shape*>* MainWindow::renderShapes [private]
```

Pointer to the vector of shapes currently being rendered.

8.8.4.8 shapeTable

```
QTableWidget* MainWindow::shapeTable [private]
```

Pointer to the table widget used to display shape properties in the report.

8.8.4.9 sortDropdown

```
QComboBox* MainWindow::sortDropdown [private]
```

Dropdown QComboBox for selecting the sorting criterion in the shape report.

8.8.4.10 sortOrderDropdown

```
QComboBox* MainWindow::sortOrderDropdown [private]
```

Dropdown QComboBox for selecting the sorting order (ascending/descending) in the shape report.

8.8.4.11 tabWidget

```
QTabWidget* MainWindow::tabWidget [private]
```

Pointer to the tab widget (if used for the shape report, otherwise could be for other purposes).

8.8.4.12 teamNameLabel

```
QLabel* MainWindow::teamNameLabel [private]
```

Label to display the team name in the Contact Us display.

8.8.4.13 ui

```
Ui::MainWindow* MainWindow::ui [private]
```

Pointer to the auto-generated UI class.

8.8.4.14 userStatusLabel

```
QLabel* MainWindow::userStatusLabel [private]
```

Label in the status bar to display the current user's login status.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.cpp

8.9 Parser::MorphicShape Struct Reference

Internal accumulator structure for parsing shape data from JSON.

Collaboration diagram for Parser::MorphicShape:

Public Attributes

- std::string shapeType = ""

The type of the shape (e.g., "Line", "Circle").

- int shapeId = 0

  The unique identifier for the shape type.
- int trackerId = 0

  A tracking ID for the shape instance.
- alpha::vector< int > shapeDimensions

  A vector holding the geometric dimensions of the shape.
- QPen pen = QPen()

  The QPen object defining the shape's outline properties.
- QBrush brush = QBrush()

  The QBrush object defining the shape's fill properties.
- QPoint coords = QPoint()

  The primary coordinates (e.g., top-left point) of the shape.
- QString textString

  The string content for Text shapes.
- GlobalColor textColor

  The color of the text for Text shapes.
- QFont font

  The QFont object for Text shapes.
- AlignmentFlag textAlignment

  The alignment for Text shapes.

8.9.1 Detailed Description

Internal accumulator structure for parsing shape data from JSON.

This structure temporarily holds the properties of a shape as they are parsed from a JSON object. It allows for jumbled key-value pairs and converts string values to their appropriate C++ types before a final Shape object is constructed.

8.9.2 Member Data Documentation

8.9.2.1 brush

```
QBrush Parser::MorphicShape::brush = QBrush()
```

The QBrush object defining the shape's fill properties.

8.9.2.2 coords

```
QPoint Parser::MorphicShape::coords = QPoint()
```

The primary coordinates (e.g., top-left point) of the shape.

8.9.2.3 font

```
QFont Parser::MorphicShape::font
```

The QFont object for Text shapes.

8.9.2.4 pen

```
QPen Parser::MorphicShape::pen = QPen()
```

The QPen object defining the shape's outline properties.

8.9.2.5 shapeDimensions

```
alpha::vector<int> Parser::MorphicShape::shapeDimensions
```

A vector holding the geometric dimensions of the shape.

8.9.2.6 shapeId

```
int Parser::MorphicShape::shapeId = 0
```

The unique identifier for the shape type.

8.9.2.7 shapeType

```
std::string Parser::MorphicShape::shapeType = ""
```

The type of the shape (e.g., "Line", "Circle").

8.9.2.8 textAlignment

```
AlignmentFlag Parser::MorphicShape::textAlignment
```

The alignment for Text shapes.

8.9.2.9 textColor

```
GlobalColor Parser::MorphicShape::textColor
```

The color of the text for Text shapes.

8.9.2.10 textString

```
QString Parser::MorphicShape::textString
```

The string content for Text shapes.

8.9.2.11 trackerId

```
int Parser::MorphicShape::trackerId = 0
```

A tracking ID for the shape instance.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h

8.10 Parser Class Reference

Provides functionality for parsing JSON data to C++ objects and vice-versa.

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

Collaboration diagram for Parser:

Classes

- struct MorphicShape

  Internal accumulator structure for parsing shape data from JSON.
- struct RawUser

  Internal accumulator structure for parsing user account data from JSON.

Public Member Functions

- Parser ()=default

  Default constructor. Initializes a Parser object.
- ~Parser ()=default

  Default destructor. Cleans up resources used by the Parser object.
- Parser (const Parser &)=delete

  Deleted copy constructor to prevent copying Parser objects.
- Parser & operator= (const Parser &)=delete

  Deleted copy assignment operator to prevent copying Parser objects.
- Parser (Parser &&)=delete

  Deleted move constructor to prevent moving Parser objects.
- Parser & operator= (Parser &&)=delete

  Deleted move assignment operator to prevent moving Parser objects.
- void PrintShapeVector (const alpha::vector< Shape * > &shapes)

  Prints properties of shapes in a vector to the console.
- alpha::vector< Shape * > JsonToShapes (const std::string &json)

  Converts a JSON string representation into a vector of Shape objects.
- std::string ShapesToJson (const alpha::vector< Shape * > &shapes)

  Converts a vector of Shape pointers into a JSON string representation.
- alpha::vector< UserAccount * > JsonToUsers (const std::string &json)

  Converts a JSON string representation into a vector of UserAccount objects.
- std::string UsersToJson (const alpha::vector< UserAccount * > &users)

  Converts a vector of UserAccount pointers into a JSON string representation.

Static Public Member Functions

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

  Converts a JSON string representation into a QVector of Testimonial objects.
- static std::string TestimonialsToJson (const QVector< Testimonial > &testimonials)

  Converts a QVector of Testimonial objects into a JSON string representation.

Private Member Functions

- MorphicShape ParseJsonObject (const std::string json, size_t &index)

  Parses a single JSON object (data within '{}') into a MorphicShape.
- void UpdateAccumulator (const std::string &key, const std::string &value, MorphicShape &tempShape)

  Updates a MorphicShape accumulator with a parsed key-value pair.
- Shape * BuildShape (MorphicShape tempShape)

  Constructs a concrete Shape object from a populated MorphicShape accumulator.
- void SkipWhitespace (const std::string &json, size_t &index)

  Advances the parsing index past any whitespace characters.
- std::string ExtractKey (const std::string &json, size_t &index)

  Extracts a JSON key (a string enclosed in double quotes) from the input string.
- std::string ExtractValue (const std::string &json, size_t &index)

  Extracts a JSON value from the input string.
- std::string ExtractInteger (const std::string &json, size_t &index)

  Extracts an integer value from the JSON string.
- std::string ExtractArray (const std::string &json, size_t &index)

  Extracts a JSON array (content within '[]') as a string.

- std::string ExtractLiteral (const std::string &json, size_t &index)

  Extracts a JSON literal (true, false, null) as a string.
- alpha::vector< int > StringToVector (const std::string &value)

  Converts a string representation of a JSON array of integers into an alpha::vector<int>.
- std::string AppendCommonShapeData (const Shape *shape)

  Appends common shape properties (ID, Type, Dimensions, Pen properties) to a JSON string.
- std::string AppendBrushData (const Shape *shape)

  Appends QBrush properties (BrushColor, BrushStyle) to a JSON string.
- std::string AppendTextData (const Shape *shape)

  Appends all properties specific to Text objects to a JSON string.
- std::string GetShapeDimensions (const Shape *shape)

  Gets the geometric dimensions of a shape as a JSON array string.
- std::string GetColor (const QColor &objectColor)

  Converts a QColor object to its string representation (e.g., "red", "blue").
- std::string GetPenStyle (const Shape *shape)

  Gets the pen style of a shape as a string (e.g., "SolidLine", "DashLine").
- std::string GetPenCapStyle (const Shape *shape)

  Gets the pen cap style of a shape as a string (e.g., "FlatCap", "RoundCap").
- std::string GetPenJoinStyle (const Shape *shape)

  Gets the pen join style of a shape as a string (e.g., "MiterJoin", "BevelJoin").
- std::string GetBrushStyle (const Shape *shape)

  Gets the brush style of a shape as a string (e.g., "SolidPattern", "NoBrush").
- std::string GetAlignmentFlag (const Text *text)

  Gets the text alignment of a Text object as a string (e.g., "AlignLeft", "AlignCenter").
- std::string GetFontStyle (const Text *text)

  Gets the font style of a Text object as a string (e.g., "StyleNormal", "StyleItalic").
- std::string GetFontWeight (const Text *text)

  Gets the font weight of a Text object as a string (e.g., "Normal", "Bold").

Static Private Member Functions

- static void UpdateUserAccumulator (const std::string &key, const std::string &value, RawUser &acc)

  Updates a RawUser accumulator with a parsed key-value pair for user data.

8.10.1 Detailed Description

Provides functionality for parsing JSON data to C++ objects and vice-versa.

The Parser class handles the serialization and deserialization of various data structures used in the application, such as Shapes, UserAccounts, and Testimonials, converting them to and from JSON string representations. It includes methods for both forward parsing (JSON to object) and reverse parsing (object to JSON). This class does not store any data itself and its copy/move operations are disabled.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 Parser() [1/3]

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

Default constructor. Initializes a Parser object.

Here is the caller graph for this function:

8.10.2.2 ~Parser()

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

Default destructor. Cleans up resources used by the Parser object.

8.10.2.3 Parser() [2/3]

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

Deleted copy constructor to prevent copying Parser objects.

Here is the call graph for this function:

8.10.2.4 Parser() [3/3]

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

Deleted move constructor to prevent moving Parser objects.

Here is the call graph for this function:

8.10.3 Member Function Documentation

8.10.3.1 AppendBrushData()

```
std::string Parser::AppendBrushData (
            const Shape * shape)  [private]
```

Appends QBrush properties (BrushColor, BrushStyle) to a JSON string.

This helper function is used for shapes that have fill properties.

Parameters

shape	A constant pointer to the Shape object whose brush data is to be serialized.

Returns

A std::string containing the JSON representation of brush data.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.2 AppendCommonShapeData()

```
std::string Parser::AppendCommonShapeData (
            const Shape * shape)  [private]
```

Appends common shape properties (ID, Type, Dimensions, Pen properties) to a JSON string.

This helper function is used during the serialization of shapes to JSON. It formats 9 common key-value pairs.

Parameters

shape	A constant pointer to the Shape object to serialize.

Returns

A std::string containing the JSON representation of common shape data.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.3 AppendTextData()

```
std::string Parser::AppendTextData (
            const Shape * shape)  [private]
```

Appends all properties specific to Text objects to a JSON string.

Serializes properties like TextString, PointSize, Color, Alignment, Font.

Parameters

shape	A constant pointer to the Shape object, expected to be a Text object.

Returns

A std::string containing the JSON representation of text data.

Exceptions

std::runtime_error	If the provided shape cannot be cast to Text.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.4 BuildShape()

```
Shape * Parser::BuildShape (
            MorphicShape tempShape)  [private]
```

Constructs a concrete Shape object from a populated MorphicShape accumulator.

Based on the `shapeId` in `tempShape`, this function dynamically allocates and initializes the appropriate derived Shape object (e.g., Line, Circle).

Parameters

tempShape	The MorphicShape object containing the data for the new shape.

Returns

A pointer to the newly instantiated Shape object, or nullptr if construction fails.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.5 ExtractArray()

```
std::string Parser::ExtractArray (
            const std::string & json,
            size_t & index)  [private]
```

Extracts a JSON array (content within '[]') as a string.

Assumes `index` is at the opening bracket of the array. Modifies `index` to point after the closing bracket of the array.

Parameters

json	The JSON string being parsed.
index	A reference to the current parsing index, modified by the function.

Returns

The extracted array (including brackets) as a std::string.

Exceptions

std::runtime_error	If the closing bracket is missing.

Here is the caller graph for this function:

8.10.3.6 ExtractInteger()

```
std::string Parser::ExtractInteger (
            const std::string & json,
            size_t & index)  [private]
```

Extracts an integer value from the JSON string.

Assumes `index` is at the first digit of the integer. Modifies `index` to point after the last digit of the integer.

Parameters

json	The JSON string being parsed.
index	A reference to the current parsing index, modified by the function.

Returns

The extracted integer as a std::string.

Here is the caller graph for this function:

8.10.3.7 ExtractKey()

```
std::string Parser::ExtractKey (
            const std::string & json,
            size_t & index)  [private]
```

Extracts a JSON key (a string enclosed in double quotes) from the input string.

Assumes `index` is at the opening double quote of the key. Modifies `index` to point immediately after the closing double quote of the key.

Parameters

json	The JSON string being parsed.
index	A reference to the current parsing index, modified by the function.

Returns

The extracted key as a std::string.

Here is the caller graph for this function:

8.10.3.8 ExtractLiteral()

```
std::string Parser::ExtractLiteral (
            const std::string & json,
            size_t & index)  [private]
```

Extracts a JSON literal (true, false, null) as a string.

Assumes `index` is at the first character of the literal. Modifies `index` to point after the last character of the literal.

Parameters

json	The JSON string being parsed.
index	A reference to the current parsing index, modified by the function.

Returns

The extracted literal as a std::string.

Here is the caller graph for this function:

8.10.3.9 ExtractValue()

```
std::string Parser::ExtractValue (
            const std::string & json,
            size_t & index)  [private]
```

Extracts a JSON value from the input string.

Handles strings, numbers, arrays, and boolean literals. Modifies `index` to point after the extracted value.

Parameters

json	The JSON string being parsed.
index	A reference to the current parsing index, modified by the function.

Returns

The extracted value as a std::string.

Exceptions

std::runtime_error	If the value type is unexpected.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.10 GetAlignmentFlag()

```
std::string Parser::GetAlignmentFlag (
            const Text * text)  [private]
```

Gets the text alignment of a Text object as a string (e.g., "AlignLeft", "AlignCenter").

Parameters

text	A constant pointer to the Text object.

Returns

A std::string representing the text alignment.

Exceptions

std::runtime_error	If the alignment flag is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.11 GetBrushStyle()

```
std::string Parser::GetBrushStyle (
            const Shape * shape)  [private]
```

Gets the brush style of a shape as a string (e.g., "SolidPattern", "NoBrush").

Parameters

shape	A constant pointer to the Shape object.

Returns

A std::string representing the brush style.

Exceptions

std::runtime_error	If the brush style is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.12 GetColor()

```
std::string Parser::GetColor (
            const QColor & objectColor)  [private]
```

Converts a QColor object to its string representation (e.g., "red", "blue").

Parameters

objectColor	The QColor to be converted.

Returns

A std::string representing the color name.

Exceptions

std::runtime_error	If the color is not one of the predefined known colors.

Here is the caller graph for this function:

8.10.3.13 GetFontStyle()

```
std::string Parser::GetFontStyle (
            const Text * text)  [private]
```

Gets the font style of a Text object as a string (e.g., "StyleNormal", "StyleItalic").

Parameters

text	A constant pointer to the Text object.

Returns

A std::string representing the font style.

Exceptions

std::runtime_error	If the font style is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.14 GetFontWeight()

```
std::string Parser::GetFontWeight (
            const Text * text)  [private]
```

Gets the font weight of a Text object as a string (e.g., "Normal", "Bold").

Parameters

text	A constant pointer to the Text object.

Returns

A std::string representing the font weight.

Exceptions

std::runtime_error	If the font weight is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.15 GetPenCapStyle()

```
std::string Parser::GetPenCapStyle (
            const Shape * shape)  [private]
```

Gets the pen cap style of a shape as a string (e.g., "FlatCap", "RoundCap").

Parameters

shape	A constant pointer to the Shape object.

Returns

A std::string representing the pen cap style.

Exceptions

std::runtime_error	If the pen cap style is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.16 GetPenJoinStyle()

```
std::string Parser::GetPenJoinStyle (
            const Shape * shape)  [private]
```

Gets the pen join style of a shape as a string (e.g., "MiterJoin", "BevelJoin").

Parameters

shape	A constant pointer to the Shape object.

Returns

A std::string representing the pen join style.

Exceptions

std::runtime_error	If the pen join style is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.17 GetPenStyle()

```
std::string Parser::GetPenStyle (
            const Shape * shape)  [private]
```

Gets the pen style of a shape as a string (e.g., "SolidLine", "DashLine").

Parameters

shape	A constant pointer to the Shape object.

Returns

A std::string representing the pen style.

Exceptions

std::runtime_error	If the pen style is unknown.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.18 GetShapeDimensions()

```
std::string Parser::GetShapeDimensions (
            const Shape * shape)  [private]
```

Gets the geometric dimensions of a shape as a JSON array string.

The specific dimensions depend on the shape type (e.g., coordinates for Line, points for Polyline/Polygon, x,y,length,width for Rectangle).

Parameters

shape	A constant pointer to the Shape object.

Returns

A std::string representing a JSON array of the shape's dimensions.

Exceptions

std::runtime_error	If the shape type is unknown or casting fails.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.19 JsonToShapes()

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

Converts a JSON string representation into a vector of Shape objects.

This is a forward parsing method that takes a JSON string, parses it, and constructs a vector of dynamically allocated Shape objects.

Parameters

json	A constant reference to a string containing the JSON data for shapes.

Returns

An alpha::vector of Shape pointers, each pointing to an instantiated Shape object.

Exceptions

std::runtime_error	If the JSON string is malformed or parsing fails.

Here is the call graph for this function:

8.10.3.20 JsonToTestimonials()

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

Converts a JSON string representation into a QVector of Testimonial objects.

This static forward parsing method uses Qt's JSON utilities to parse testimonials.

Parameters

json	A constant reference to a string containing the JSON data for testimonials.

Returns

A QVector of Testimonial objects.

Exceptions

std::runtime_error	If JSON parsing fails or the top-level structure is not an array.

Here is the call graph for this function:

8.10.3.21 JsonToUsers()

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

Converts a JSON string representation into a vector of UserAccount objects.

This forward parsing method processes a JSON string to create UserAccount objects.

Parameters

json	A constant reference to a string containing the JSON data for user accounts.

Returns

An alpha::vector of UserAccount pointers.

Exceptions

std::runtime_error	If the JSON string is malformed or essential user data is missing.

Here is the call graph for this function:

8.10.3.22 operator=() [1/2]

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

Deleted copy assignment operator to prevent copying Parser objects.

Here is the call graph for this function:

8.10.3.23 operator=() [2/2]

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

Deleted move assignment operator to prevent moving Parser objects.

Here is the call graph for this function:

8.10.3.24 ParseJsonObject()

```
Parser::MorphicShape Parser::ParseJsonObject (
            const std::string json,
            size_t & index)  [private]
```

Parses a single JSON object (data within '{}') into a MorphicShape.

This helper function processes one JSON object from the input string, extracting key-value pairs and populating a MorphicShape accumulator.

Parameters

json	The JSON string being parsed.
index	A reference to the current parsing index within the JSON string, modified by the function.

Returns

A MorphicShape object containing the parsed data.

Exceptions

std::runtime_error	If the JSON object is malformed.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.25 PrintShapeVector()

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

Prints properties of shapes in a vector to the console.

Iterates through a vector of Shape pointers and prints their ID, TrackerId, and ShapeType to standard output. Primarily for debugging purposes.

Parameters

shapes	A constant reference to an alpha::vector of Shape pointers.

8.10.3.26 ShapesToJson()

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

Converts a vector of Shape pointers into a JSON string representation.

This is a reverse parsing method that takes a vector of Shape objects and serializes them into a JSON formatted string.

Parameters

shapes	A constant reference to an alpha::vector of Shape pointers.

Returns

A string formatted in JSON containing all key:value pairs for the shapes.

Here is the call graph for this function:

8.10.3.27 SkipWhitespace()

```
void Parser::SkipWhitespace (
            const std::string & json,
            size_t & index)  [private]
```

Advances the parsing index past any whitespace characters.

Modifies `index` to point to the next non-whitespace character in the `json` string.

Parameters

json	The JSON string being parsed.

index	A reference to the current parsing index, modified by the function.

Here is the caller graph for this function:

8.10.3.28 StringToVector()

```
alpha::vector< int > Parser::StringToVector (
            const std::string & value)  [private]
```

Converts a string representation of a JSON array of integers into an alpha::vector<int>.

Parses a string like "[20, 32, 41, 64]" into a vector of integers.

Parameters

value	The string representation of the integer array.

Returns

An alpha::vector<int> containing the parsed integers.

Exceptions

std::runtime_error	If the string format is invalid.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.29 TestimonialsToJson()

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

Converts a QVector of Testimonial objects into a JSON string representation.

This static reverse parsing method uses Qt's JSON utilities to serialize testimonials.

Parameters

testimonials	A constant reference to a QVector of Testimonial objects.

Returns

A string formatted in JSON representing the testimonials.

Here is the caller graph for this function:

8.10.3.30 UpdateAccumulator()

```
void Parser::UpdateAccumulator (
            const std::string & key,
            const std::string & value,
            MorphicShape & tempShape)  [private]
```

Updates a MorphicShape accumulator with a parsed key-value pair.

Converts the string `value` to its appropriate C++ type based on the `key` and stores it in the corresponding member of the `tempShape` accumulator.

Parameters

key	The JSON key as a string.
value	The JSON value as a string.
tempShape	A reference to the MorphicShape accumulator to be updated.

Exceptions

std::runtime_error	If the key is unknown or the value is invalid for the key.

Here is the call graph for this function:

Here is the caller graph for this function:

8.10.3.31 UpdateUserAccumulator()

```
void Parser::UpdateUserAccumulator (
            const std::string & key,
            const std::string & value,
            RawUser & acc)  [static], [private]
```

Updates a RawUser accumulator with a parsed key-value pair for user data.

Converts the string `value` to its appropriate C++ type based on the `key` (username, password, admin) and stores it in the `acc` accumulator. Also sets flags in `acc` to indicate which fields were found.

Parameters

key	The JSON key as a string.
value	The JSON value as a string.
acc	A reference to the RawUser accumulator to be updated.

Exceptions

std::runtime_error	If the value for 'admin' is not "true" or "false".

Here is the caller graph for this function:

8.10.3.32 UsersToJson()

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

Converts a vector of UserAccount pointers into a JSON string representation.

This reverse parsing method serializes UserAccount objects into a JSON string.

Parameters

users	A constant reference to an alpha::vector of UserAccount pointers.

Returns

A string formatted in JSON representing the user accounts.

Here is the call graph for this function:

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.cpp

8.11 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 (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- QPolygon pointsList

  list of points

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 Detailed Description

The Polygon class.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 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:

8.11.3 Member Function Documentation

8.11.3.1 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:

8.11.3.2 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:

8.11.3.3 getPointsList()

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

getPointsList - Returns the pointsList by value

Returns

Here is the caller graph for this function:

8.11.3.4 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.

8.11.3.5 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:

8.11.3.6 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:

8.11.3.7 setPointsList()

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

Mutator Functions.

setPointsList - Sets the pointsList to the passed newPointsList

Parameters

newPointsList	- New list of points for pointsList to take

Here is the caller graph for this function:

8.11.3.8 setX()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.11.3.9 setY()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.11.4 Member Data Documentation

8.11.4.1 pointsList

```
QPolygon Polygon::pointsList  [private]
```

list of points

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.cpp

8.12 Polyline Class Reference

The Polyline class.

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

Inheritance diagram for Polyline:

Collaboration diagram for Polyline:

Public Member Functions

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

  Polyline.
- void Draw (QWidget ∗renderArea) override

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

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

- double Perimeter () const override

  Perimeter - Returns the perimater of the Polyline.
- double Area () const override

  Area - Returns 0 (Needs to have implementation to avoid being seen as virtual.
- bool isPointInside (const QPoint &point) const override

  isPointInside - Returns True if point is inside the Polyline
- 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 (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- QPolygon pointsList

  list of points

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

8.12.1 Detailed Description

The Polyline class.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 Polyline()

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

Polyline.

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 Polyline

Here is the call graph for this function:

8.12.3 Member Function Documentation

8.12.3.1 Area()

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

Area - Returns 0 (Needs to have implementation to avoid being seen as virtual.

Returns

Implements Shape.

8.12.3.2 Draw()

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

Draw - Draws a Polyline at the assigne coords with points in the pointsList.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

Here is the call graph for this function:

8.12.3.3 getPointsList()

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

getPointsList - Returns the pointsList by value

Returns

Here is the caller graph for this function:

8.12.3.4 isPointInside()

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

isPointInside - Returns True if point is inside the Polyline

Parameters

point	- points being read

Returns

Implements Shape.

8.12.3.5 Move()

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

Move - Moves the Polyline 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:

8.12.3.6 Perimeter()

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

Perimeter - Returns the perimater of the Polyline.

Returns

Implements Shape.

8.12.3.7 setPointsList()

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

Mutator Functions.

setPointsList - Sets the pointsList to the passed newPointsList

Parameters

newPointsList	- New list of points for pointsList to take

Here is the caller graph for this function:

8.12.3.8 setX()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.12.3.9 setY()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.12.4 Member Data Documentation

8.12.4.1 pointsList

```
QPolygon Polyline::pointsList  [private]
```

list of points

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.cpp

8.13 QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11User↩ ManagerE_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:

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_UserManager.cpp

8.14 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:

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_ApiClient.cpp

8.15 Parser::RawUser Struct Reference

Internal accumulator structure for parsing user account data from JSON.

Collaboration diagram for Parser::RawUser:

Public Attributes

- QString username

  The username of the user.
- QString password

  The password of the user.
- bool admin = false

  Flag indicating if the user has admin privileges.
- bool hasUsername = false

  Flag indicating if the username was found during parsing.
- bool hasPassword = false

  Flag indicating if the password was found during parsing.
- bool hasAdmin = false

  Flag indicating if the admin status was found during parsing.

8.15.1 Detailed Description

Internal accumulator structure for parsing user account data from JSON.

This structure temporarily holds the properties of a user account as they are parsed from a JSON object, along with flags to track if fields were found.

8.15.2 Member Data Documentation

8.15.2.1 admin

```
bool Parser::RawUser::admin = false
```

Flag indicating if the user has admin privileges.

8.15.2.2 hasAdmin

```
bool Parser::RawUser::hasAdmin = false
```

Flag indicating if the admin status was found during parsing.

8.15.2.3 hasPassword

```
bool Parser::RawUser::hasPassword = false
```

Flag indicating if the password was found during parsing.

8.15.2.4 hasUsername

```
bool Parser::RawUser::hasUsername = false
```

Flag indicating if the username was found during parsing.

8.15.2.5 password

```
QString Parser::RawUser::password
```

The password of the user.

8.15.2.6 username

```
QString Parser::RawUser::username
```

The username of the user.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h

8.16 Rectangle Class Reference

The Rectangle class.

```
#include "objects/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)

  Rectangle.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws a Rectangle at the assigned coords.
- double Perimeter () const override

Perimeter - Returns the perimeter of the Rectangle.

- double Area () const override

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

  isPointInside - Returns True if point is inside the Rectangle
- int getLength () const

  Accessor Functions.
- int getWidth () const
- void setLength (int newLength)

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

Public Member Functions inherited from Shape

- Shape (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- int length

  length of the rectangle
- int width

  width of the rectangle

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

8.16.1 Detailed Description

The Rectangle class.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 Rectangle()

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

Rectangle.

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
length	- Length of the Rectangle
width	- Width of the Rectangle

Here is the call graph for this function:

8.16.3 Member Function Documentation

8.16.3.1 Area()

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

Area - Returns the area of the Rectangle.

Returns

Implements Shape.

8.16.3.2 Draw()

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

Draw - Draws a Rectangle at the assigned coords.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

Here is the call graph for this function:

8.16.3.3 getLength()

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

Accessor Functions.

Here is the caller graph for this function:

8.16.3.4 getWidth()

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

Here is the caller graph for this function:

8.16.3.5 isPointInside()

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

isPointInside - Returns True if point is inside the Rectangle

Parameters

point	- point being read

Returns

Implements Shape.

Here is the call graph for this function:

8.16.3.6 Perimeter()

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

Perimeter - Returns the perimeter of the Rectangle.

Returns

Implements Shape.

8.16.3.7 setLength()

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

Mutator Functions.

Here is the caller graph for this function:

8.16.3.8 setWidth()

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

Here is the caller graph for this function:

8.16.3.9 setX()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.16.3.10 setY()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.16.4 Member Data Documentation

8.16.4.1 length

```
int Rectangle::length [private]
```

length of the rectangle

8.16.4.2 width

```
int Rectangle::width [private]
```

width of the rectangle

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.cpp

8.17 RenderArea Class Reference

The RenderArea class provides a canvas for drawing 2D shapes.

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

Inheritance diagram for RenderArea:

Collaboration diagram for RenderArea:

Public Member Functions

- RenderArea (QWidget *parent=nullptr)

  RenderArea Constructor.

Core Interaction and Display Management

Handles user mouse input for selecting and manipulating shapes, manages shape selection state, and updates display coordinates.

- void mousePressEvent (QMouseEvent ∗event) override

  Handles mouse press events to detect clicks on shapes for selection.
- void mouseMoveEvent (QMouseEvent ∗event) override

  Handles mouse move events to drag selected shapes if editing is allowed.
- void mouseDoubleClickEvent (QMouseEvent ∗event) override

  Handles mouse double-click events, e.g., for expanding shape details.
- void mouseReleaseEvent (QMouseEvent ∗event) override

  Handles mouse release events to finalize a shape move operation.
- void resetSelection ()

  Resets the selection state of all shapes, clearing the current selection.
- void updateShapeDisplayCoords (Shape ∗item, const QPoint &position) const

  Updates display coordinates of a shape's properties, typically in an associated view.

Accessor Functions

Getters for RenderArea properties and state.

- const alpha::vector< Shape ∗ > & getShapes () const

  Returns a const reference to the vector of shapes being rendered.
- int getShapeSelected () const

  Returns the tracker ID of the currently selected shape, or -1 if no shape is selected.
- int getShapeSelectedIndex () const

  Returns the index of the currently selected shape in the `renderShapes` vector, or -1 if none.

Mutator Functions

Setters for RenderArea properties and state.

- void setRenderShapes (const alpha::vector< Shape ∗ > ∗renderShapes)

  Sets the vector of shapes to be rendered. The RenderArea does not own these shapes.
- void setShapeSelectedIndex (int newIndex)

  Sets the index of the currently selected shape.
- void setEditPrivileges (bool edit)

  Sets whether shapes can be edited (e.g., moved) on the canvas.

Protected Member Functions

- void paintEvent (QPaintEvent ∗event) override

  Handles paint events to draw all shapes onto the widget.

Private Attributes

Private Members

Internal data members for managing state and rendered content.

- const alpha::vector< Shape ∗ > ∗ renderShapes

  Pointer to the external vector holding shapes to be rendered.
- int shapeSelectedIndex

  Index of the currently selected shape in `renderShapes`. -1 if no shape is selected. This ensures only one shape can be selected at a time.
- bool allowEditing = false

  Flag indicating whether shapes can be moved/edited. Defaults to false.

8.17.1 Detailed Description

The RenderArea class provides a canvas for drawing 2D shapes.

This widget handles mouse events for selecting and moving shapes, and it uses QPainter to render shapes provided to it. It also manages the selection state of shapes and editing privileges.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 RenderArea()

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

RenderArea Constructor.

Parameters

parent	The parent QWidget, if any.

8.17.3 Member Function Documentation

8.17.3.1 getShapes()

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

Returns a const reference to the vector of shapes being rendered.

8.17.3.2 getShapeSelected()

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

Returns the tracker ID of the currently selected shape, or -1 if no shape is selected.

8.17.3.3 getShapeSelectedIndex()

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

Returns the index of the currently selected shape in the `renderShapes` vector, or -1 if none.

8.17.3.4 mouseDoubleClickEvent()

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

Handles mouse double-click events, e.g., for expanding shape details.

8.17.3.5 mouseMoveEvent()

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

Handles mouse move events to drag selected shapes if editing is allowed.

Here is the call graph for this function:

8.17.3.6 mousePressEvent()

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

Handles mouse press events to detect clicks on shapes for selection.

8.17.3.7 mouseReleaseEvent()

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

Handles mouse release events to finalize a shape move operation.

Here is the call graph for this function:

8.17.3.8 paintEvent()

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

Handles paint events to draw all shapes onto the widget.

Iterates through the `renderShapes` vector and calls the `Draw` method for each shape.

Parameters

event	The QPaintEvent object.

8.17.3.9 resetSelection()

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

Resets the selection state of all shapes, clearing the current selection.

8.17.3.10 setEditPrivileges()

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

Sets whether shapes can be edited (e.g., moved) on the canvas.

8.17.3.11 setRenderShapes()

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

Sets the vector of shapes to be rendered. The RenderArea does not own these shapes.

8.17.3.12 setShapeSelectedIndex()

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

Sets the index of the currently selected shape.

8.17.3.13 updateShapeDisplayCoords()

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

Updates display coordinates of a shape's properties, typically in an associated view.

Here is the call graph for this function:

Here is the caller graph for this function:

8.17.4 Member Data Documentation

8.17.4.1 allowEditing

```
bool RenderArea::allowEditing = false  [private]
```

Flag indicating whether shapes can be moved/edited. Defaults to false.

8.17.4.2 renderShapes

```
const alpha::vector<Shape*>* RenderArea::renderShapes [private]
```

Pointer to the external vector holding shapes to be rendered.

8.17.4.3 shapeSelectedIndex

```
int RenderArea::shapeSelectedIndex [private]
```

Index of the currently selected shape in `renderShapes`. -1 if no shape is selected. This ensures only one shape can be selected at a time.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.cpp

8.18 RenderAreaManager Class Reference

Manages the shapes to be rendered and interacts with the backend API.

```
#include <RenderAreaManager.h>
```

Inheritance diagram for RenderAreaManager:

Collaboration diagram for RenderAreaManager:

Signals

Render Area Signals

Signals emitted by RenderAreaManager to communicate with other parts of the application.

These signals are primarily used to notify the UI about changes in the shape data or to convey status messages.

- void renderAreaChanged ()

Emitted when the render area has changed and needs redrawing.

- void renderAreaNotChanged (const QString &message)

  Emitted when an operation completes without changing the render area, often with a message.
- void statusMessage (const QString &message)

  Emitted to provide a status message to the user.

Public Member Functions

Core Management Functions

Public methods for RenderAreaManager.

This section includes constructors, destructors, and primary interface functions for managing and accessing shape data.

- RenderAreaManager (QObject *parent=nullptr)

  Constructs a RenderAreaManager object and connects API client signals.
- ~RenderAreaManager ()

  Destroys the RenderAreaManager object and deallocates rendered shapes.
- alpha::vector< Shape * > * getShapesRef ()

  Retrieves a pointer to the vector of rendered shapes.

Shape Manipulation

Functions for manipulating shapes within the RenderAreaManager.

These methods handle the creation, modification, and deletion of shapes. Most of these operations will trigger a signal to update the rendering area and will attempt to save the changes to the backend.

- void addShape (Shape *shape)

  Adds a new shape to the rendering list and triggers a save.
- void modifyShape (Shape *shape, QString key, int value)

  Modifies a property of an existing shape based on a key-value pair.
- void modifyDisplayedText (Text *obj, QString newText)

  Modifies the displayed text of a Text shape.
- void deleteShape (const int trackerId)

  Deletes a shape identified by its tracker ID.
- void deleteAllShapes ()

  Requests the deletion of all shapes from the backend.
- void loadShapes ()

  Initiates loading shapes from the backend webservice.
- void saveShapes ()

  Initiates saving the current shapes to the backend webservice.

Private Slots

API Client Response Handlers

Private slots for handling responses from the ApiClient.

These slots are connected to signals from the ApiClient and process the results of asynchronous network operations (GET, POST, DELETE).

- void onGoodGetResponse (const QString &json)

  Slot to handle successful GET responses from the API client.
- void onBadGetResponse (const QString &errorMsg)

  Slot to handle unsuccessful GET responses from the API client.
- void onGoodPostResponse ()

  Slot to handle successful POST responses from the API client.
- void onBadPostResponse (const QString &errorMsg)

  Slot to handle unsuccessful POST responses from the API client.
- void onGoodDeleteResponse ()

  Slot to handle successful DELETE responses from the API client.
- void onBadDeleteResponse (const QString &errorMsg)

  Slot to handle unsuccessful DELETE responses from the API client.

Private Attributes

Internal Data and Utilities

Private members for storing shape data and utility objects.

This section includes the storage for shapes and instances of helper classes like ApiClient and Parser.

- alpha::vector< Shape * > renderedShapes
  Vector storing pointers to all shapes currently managed.
- ApiClient client
  Client for making API requests to the backend.
- Parser parse
  Parser for converting shapes to/from JSON.

8.18.1 Detailed Description

Manages the shapes to be rendered and interacts with the backend API.

This class is responsible for adding, modifying, deleting, loading, and saving shapes. It communicates with an ApiClient to persist shape data and uses a Parser for serialization and deserialization. It also emits signals when the render area needs to be updated or when status messages should be displayed.

8.18.2 Constructor & Destructor Documentation

8.18.2.1 RenderAreaManager()

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

Constructs a RenderAreaManager object and connects API client signals.

Here is the call graph for this function:

8.18.2.2 ~RenderAreaManager()

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

Destroys the RenderAreaManager object and deallocates rendered shapes.

8.18.3 Member Function Documentation

8.18.3.1 addShape()

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

Adds a new shape to the rendering list and triggers a save.

Here is the call graph for this function:

8.18.3.2 deleteAllShapes()

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

Requests the deletion of all shapes from the backend.

Here is the call graph for this function:

8.18.3.3 deleteShape()

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

Deletes a shape identified by its tracker ID.

Here is the call graph for this function:

8.18.3.4 getShapesRef()

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

Retrieves a pointer to the vector of rendered shapes.

8.18.3.5 loadShapes()

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

Initiates loading shapes from the backend webservice.

8.18.3.6 modifyDisplayedText()

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

Modifies the displayed text of a Text shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.7 modifyShape()

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

Modifies a property of an existing shape based on a key-value pair.

Here is the call graph for this function:

8.18.3.8 onBadDeleteResponse

```
void RenderAreaManager::onBadDeleteResponse (
            const QString & errorMsg)  [private], [slot]
```

Slot to handle unsuccessful DELETE responses from the API client.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.9 onBadGetResponse

```
void RenderAreaManager::onBadGetResponse (
            const QString & errorMsg)  [private], [slot]
```

Slot to handle unsuccessful GET responses from the API client.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.10 onBadPostResponse

```
void RenderAreaManager::onBadPostResponse (
            const QString & errorMsg)  [private], [slot]
```

Slot to handle unsuccessful POST responses from the API client.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.11 onGoodDeleteResponse

```
void RenderAreaManager::onGoodDeleteResponse ()  [private], [slot]
```

Slot to handle successful DELETE responses from the API client.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.12 onGoodGetResponse

```
void RenderAreaManager::onGoodGetResponse (
            const QString & json)  [private], [slot]
```

Slot to handle successful GET responses from the API client.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.13 onGoodPostResponse

```
void RenderAreaManager::onGoodPostResponse () [private], [slot]
```

Slot to handle successful POST responses from the API client.

Here is the call graph for this function:

Here is the caller graph for this function:

8.18.3.14 renderAreaChanged

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

Emitted when the render area has changed and needs redrawing.

Here is the caller graph for this function:

8.18.3.15 renderAreaNotChanged

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

Emitted when an operation completes without changing the render area, often with a message.

Here is the caller graph for this function:

8.18.3.16 saveShapes()

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

Initiates saving the current shapes to the backend webservice.

Here is the caller graph for this function:

8.18.3.17 statusMessage

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

Emitted to provide a status message to the user.

Here is the caller graph for this function:

8.18.4 Member Data Documentation

8.18.4.1 client

```
ApiClient RenderAreaManager::client  [private]
```

Client for making API requests to the backend.

8.18.4.2 parse

```
Parser RenderAreaManager::parse [private]
```

Parser for converting shapes to/from JSON.

8.18.4.3 renderedShapes

```
alpha::vector<Shape*> RenderAreaManager::renderedShapes [private]
```

Vector storing pointers to all shapes currently managed.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.cpp

8.19 Shape Interface Reference

The Shape Abstract Base Class.

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

Inheritance diagram for Shape:

Collaboration diagram for Shape:

Public Member Functions

- Shape (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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

Private Member Functions

- Shape (Shape &shape)=delete
- Shape & operator= (Shape &object)=delete

Private Attributes

- const int shapeId

  int representing the id of a shape, each shapeType is given one id, Line = 1, Polyline = 2, etc.
- int trackerId

  int representing the unique id of each shape, each individual shape has its own trackerId
- string shapeType

  string representing the type of shape, (Line, Circle, etc)
- QPen pen

  QPen for the outline.
- QBrush brush

  QBrush for the fill.
- QPoint coords

  QPoint for the coordinates of the shape.
- QPainter painter

  QPainter used to paint the shape onto a rendering area.
- bool isSelected = false

  Shows the state of the shape, if it is currently selected by the user.

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

8.19.1 Detailed Description

The Shape Abstract Base Class.

8.19.2 Constructor & Destructor Documentation

8.19.2.1 Shape() [1/2]

```
Shape::Shape (
            int shapeId,
            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:

8.19.2.2 ~Shape()

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

~Shape Destructor

8.19.2.3 Shape() [2/2]

```
Shape::Shape (
            Shape & shape)  [private], [delete]
```

Here is the call graph for this function:

8.19.3 Member Function Documentation

8.19.3.1 allocateTrackerId()

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

Here is the caller graph for this function:

8.19.3.2 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.

Here is the caller graph for this function:

8.19.3.3 CreateBrushChild()

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

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

Here is the caller graph for this function:

8.19.3.4 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:

8.19.3.5 CreatePenChild()

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

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

Here is the caller graph for this function:

8.19.3.6 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:

8.19.3.7 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.

8.19.3.8 getBrush()

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

Here is the caller graph for this function:

8.19.3.9 getBrushColor()

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

Here is the caller graph for this function:

8.19.3.10 getBrushItems()

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

Here is the caller graph for this function:

8.19.3.11 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:

8.19.3.12 getBrushStyle()

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

Here is the caller graph for this function:

8.19.3.13 getChildEnd()

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

Here is the caller graph for this function:

8.19.3.14 getChildItems()

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

Here is the caller graph for this function:

8.19.3.15 getPainter()

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

Here is the caller graph for this function:

8.19.3.16 getParentItem()

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

Here is the caller graph for this function:

8.19.3.17 getPen()

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

Here is the caller graph for this function:

8.19.3.18 getPenCapStyle()

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

Here is the caller graph for this function:

8.19.3.19 getPenColor()

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

Here is the caller graph for this function:

8.19.3.20 getPenItems()

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

Here is the caller graph for this function:

8.19.3.21 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:

8.19.3.22 getPenJoinStyle()

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

Here is the caller graph for this function:

8.19.3.23 getPenStyle()

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

Here is the caller graph for this function:

8.19.3.24 getPenWidth()

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

Here is the caller graph for this function:

8.19.3.25 getPoints()

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

Here is the caller graph for this function:

8.19.3.26 getPointsItems()

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

Here is the caller graph for this function:

8.19.3.27 getSelected()

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

Here is the caller graph for this function:

8.19.3.28 getShapeId()

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

Accessor Functions - Returns the data named after them.

Here is the caller graph for this function:

8.19.3.29 getShapeType()

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

Here is the caller graph for this function:

8.19.3.30 getTrackerId()

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

Here is the caller graph for this function:

8.19.3.31 getX()

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

Here is the caller graph for this function:

8.19.3.32 getY()

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

Here is the caller graph for this function:

8.19.3.33 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.

8.19.3.34 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:

8.19.3.35 operator=()

```
Shape & Shape::operator= (
            Shape & object)  [private], [delete]
```

Here is the call graph for this function:

8.19.3.36 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.

Here is the caller graph for this function:

8.19.3.37 setBrush()

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

8.19.3.38 setInternalBrush()

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

Here is the caller graph for this function:

8.19.3.39 setInternalPen()

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

Here is the caller graph for this function:

8.19.3.40 setPen()

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

8.19.3.41 setSelected()

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

8.19.3.42 setShapeType()

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

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

8.19.3.43 setTrackerId()

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

Here is the call graph for this function:

Here is the caller graph for this function:

8.19.3.44 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:

8.19.3.45 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:

8.19.4 Friends And Related Symbol Documentation

8.19.4.1 operator<

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

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

Parameters

shape1	
shape2	

Returns

8.19.4.2 operator==

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

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

Parameters

shape1	
shape2	

Returns

8.19.5 Member Data Documentation

8.19.5.1 brush

```
QBrush Shape::brush [private]
```

QBrush for the fill.

8.19.5.2 brushItems

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

vector of QTreeWidgetItem∗ holding data of all brush items

8.19.5.3 childItems

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

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

8.19.5.4 coords

```
QPoint Shape::coords [private]
```

QPoint for the coordinates of the shape.

8.19.5.5 isSelected

```
bool Shape::isSelected = false [private]
```

Shows the state of the shape, if it is currently selected by the user.

8.19.5.6 nextTracker

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

8.19.5.7 painter

```
QPainter Shape::painter [private]
```

QPainter used to paint the shape onto a rendering area.

8.19.5.8 parentItem

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

Mutator Functions.

QTreeWidgetItem∗ holding treeWidget data of each shape

8.19.5.9 pen

```
QPen Shape::pen [private]
```

QPen for the outline.

8.19.5.10 penItems

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

vector of QTreeWidgetItem∗ holding data of all pen items

8.19.5.11 pointsItems

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

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

8.19.5.12 shapeId

```
const int Shape::shapeId [private]
```

int representing the id of a shape, each shapeType is given one id, Line = 1, Polyline = 2, etc.

8.19.5.13 shapeType

```
string Shape::shapeType [private]
```

string representing the type of shape, (Line, Circle, etc)

8.19.5.14 trackerId

```
int Shape::trackerId [private]
```

int representing the unique id of each shape, each individual shape has its own trackerId

8.19.5.15 trackersInUse

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

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.cpp

8.20 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 ()

Private Slots

- void onGoodGetResponse (const QString &json)
- void onBadGetResponse (const QString &errorMsg)
- void onGoodPostResponse ()
- void onBadPostResponse (const QString &errorMsg)
- void onGoodDeleteResponse ()
- void onBadDeleteResponse (const QString &errorMsg)

Private Attributes

- alpha::vector< Shape * > shapes
- ApiClient client
- Parser parse

8.20.1 Constructor & Destructor Documentation

8.20.1.1 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:

8.20.1.2 ~ShapesManager()

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

8.20.2 Member Function Documentation

8.20.2.1 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:

8.20.2.2 deleteAllShapes()

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

Here is the call graph for this function:

8.20.2.3 deleteShape()

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

Here is the call graph for this function:

8.20.2.4 getShapesRef()

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

Returns the shapes vector as a reference.

8.20.2.5 loadShapes()

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

8.20.2.6 modifyShape()

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

Here is the call graph for this function:

8.20.2.7 onBadDeleteResponse

```
void ShapesManager::onBadDeleteResponse (
            const QString & errorMsg)  [private], [slot]
```

Here is the call graph for this function:

Here is the caller graph for this function:

8.20.2.8 onBadGetResponse

```
void ShapesManager::onBadGetResponse (
            const QString & errorMsg)  [private], [slot]
```

Here is the call graph for this function:

Here is the caller graph for this function:

8.20.2.9 onBadPostResponse

```
void ShapesManager::onBadPostResponse (
            const QString & errorMsg)  [private], [slot]
```

Here is the call graph for this function:

Here is the caller graph for this function:

8.20.2.10 onGoodDeleteResponse

```
void ShapesManager::onGoodDeleteResponse () [private], [slot]
```

Here is the call graph for this function:

Here is the caller graph for this function:

8.20.2.11 onGoodGetResponse

```
void ShapesManager::onGoodGetResponse (
            const QString & json) [private], [slot]
```

These slot functions receive signals from the ApiClient class and hold the code on what to do next after one of these responses from the webservice. Here is the call graph for this function:

Here is the caller graph for this function:

8.20.2.12 onGoodPostResponse

```
void ShapesManager::onGoodPostResponse () [private], [slot]
```

Here is the call graph for this function:

Here is the caller graph for this function:

8.20.2.13 saveShapes()

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

Here is the caller graph for this function:

8.20.2.14 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:

8.20.2.15 shapesNotChanged

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

Here is the caller graph for this function:

8.20.2.16 statusMessage

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

Here is the caller graph for this function:

8.20.3 Member Data Documentation

8.20.3.1 client

```
ApiClient ShapesManager::client  [private]
```

8.20.3.2 parse

```
Parser ShapesManager::parse  [private]
```

8.20.3.3 shapes

```
alpha::vector<Shape*> ShapesManager::shapes  [private]
```

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.cpp

8.21 Square Class Reference

The Square class.

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

Inheritance diagram for Square:

Collaboration diagram for Square:

Public Member Functions

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

  Square.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws a Square at the assigned coords.
- double Perimeter () const override

Perimeter - Returns the perimater of the Square.

- double Area () const override

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

  isPointInside - Returns True if point is inside the Square
- int getLength () const

  Accessor Functions.
- void setLength (int newLength)

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

Public Member Functions inherited from Shape

- Shape (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- int length

  side length of the Square

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

8.21.1 Detailed Description

The Square class.

8.21.2 Constructor & Destructor Documentation

8.21.2.1 Square()

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

Square.

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
length	- Side length of the Square

Here is the call graph for this function:

8.21.3 Member Function Documentation

8.21.3.1 Area()

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

Area - Returns the area of the Square.

Returns

Implements Shape.

8.21.3.2 Draw()

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

Draw - Draws a Square at the assigned coords.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

Here is the call graph for this function:

8.21.3.3 getLength()

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

Accessor Functions.

Here is the caller graph for this function:

8.21.3.4 isPointInside()

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

isPointInside - Returns True if point is inside the Square

Parameters

point	- point being read

Returns

Implements Shape.

Here is the call graph for this function:

8.21.3.5 Perimeter()

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

Perimeter - Returns the perimater of the Square.

Returns

Implements Shape.

8.21.3.6 setLength()

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

Mutator Functions.

Here is the caller graph for this function:

8.21.3.7 setX()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.21.3.8 setY()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.21.4 Member Data Documentation

8.21.4.1 length

```
int Square::length [private]
```

side length of the Square

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.cpp

8.22 Testimonial Class Reference

Represents a user testimonial.

```
#include <Testimonial.h>
```

Collaboration diagram for Testimonial:

Public Member Functions

Constructor

Constructs a Testimonial object.

Initializes a new testimonial with the provided author, content, and guest status. The timestamp is automatically set to the current date and time, and `isSatisfactory` *defaults to true.*

- Testimonial (const QString &author="", const QString &content="", bool isGuest=true)
  Constructor with optional parameters.

Getters

Public getters for accessing testimonial data.

These methods provide read-only access to the private member variables of the Testimonial object.

- QString getAuthor () const
  Returns the author of the testimonial.
- QString getContent () const
  Returns the content of the testimonial.
- QDateTime getTimestamp () const
  Returns the timestamp when the testimonial was submitted.
- bool isGuest () const
  Returns true if the author is a guest, false otherwise.
- bool isSatisfactory () const
  Returns true if the testimonial is marked as satisfactory for display.

Setters

Public setter for the satisfaction flag.

This method allows modification of the `m_isSatisfactory` flag, typically used for moderation purposes.

- void setIsSatisfactory (bool value)
  Sets the satisfactory status of the testimonial.

Private Attributes

Member Data

Private member variables storing testimonial attributes.

These variables store the core information for each testimonial.

- QString m_author
  Name of the person giving the testimonial.
- QString m_content
  Actual testimonial text.
- QDateTime m_timestamp
  Timestamp of when the testimonial was submitted.
- bool m_isGuest
  Flag indicating if the author is a guest or a registered user.
- bool m_isSatisfactory
  Flag indicating if the testimonial is deemed satisfactory for display (e.g., after moderation).

JSON Conversion

JSON conversion methods for database storage and retrieval.

These static and member methods handle the serialization of a Testimonial object to a QJsonObject and deserialization from a QJsonObject.

- QJsonObject toJson () const
  Converts the Testimonial object to a QJsonObject.
- static Testimonial fromJson (const QJsonObject &json)
  Creates a Testimonial object from a QJsonObject.

8.22.1 Detailed Description

Represents a user testimonial.

This class encapsulates the data for a testimonial, including the author's name, the content of the testimonial, a timestamp, whether the author is a guest, and a flag indicating if the testimonial is satisfactory for display. It also provides methods for converting testimonial data to and from JSON format.

8.22.2 Constructor & Destructor Documentation

8.22.2.1 Testimonial()

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

Constructor with optional parameters.

Here is the call graph for this function:

Here is the caller graph for this function:

8.22.3 Member Function Documentation

8.22.3.1 fromJson()

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

Creates a Testimonial object from a QJsonObject.

Here is the call graph for this function:

Here is the caller graph for this function:

8.22.3.2 getAuthor()

```
QString Testimonial::getAuthor () const
```

Returns the author of the testimonial.

8.22.3.3 getContent()

```
QString Testimonial::getContent () const
```

Returns the content of the testimonial.

8.22.3.4 getTimestamp()

```
QDateTime Testimonial::getTimestamp () const
```

Returns the timestamp when the testimonial was submitted.

8.22.3.5 isGuest()

```
bool Testimonial::isGuest () const
```

Returns true if the author is a guest, false otherwise.

Here is the caller graph for this function:

8.22.3.6 isSatisfactory()

```
bool Testimonial::isSatisfactory () const
```

Returns true if the testimonial is marked as satisfactory for display.

8.22.3.7 setIsSatisfactory()

```
void Testimonial::setIsSatisfactory (
            bool value)
```

Sets the satisfactory status of the testimonial.

8.22.3.8 toJson()

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

Converts the Testimonial object to a QJsonObject.

8.22.4 Member Data Documentation

8.22.4.1 m_author

```
QString Testimonial::m_author [private]
```

Name of the person giving the testimonial.

8.22.4.2 m_content

```
QString Testimonial::m_content [private]
```

Actual testimonial text.

8.22.4.3 m_isGuest

```
bool Testimonial::m_isGuest [private]
```

Flag indicating if the author is a guest or a registered user.

8.22.4.4 m_isSatisfactory

```
bool Testimonial::m_isSatisfactory [private]
```

Flag indicating if the testimonial is deemed satisfactory for display (e.g., after moderation).

8.22.4.5 m_timestamp

```
QDateTime Testimonial::m_timestamp [private]
```

Timestamp of when the testimonial was submitted.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.cpp

8.23 TestimonialDialog Class Reference

The TestimonialDialog class provides a user interface for submitting testimonials.

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

Inheritance diagram for TestimonialDialog:

Collaboration diagram for TestimonialDialog:

Public Member Functions

- TestimonialDialog (QWidget ∗parent=nullptr)

  Constructs a TestimonialDialog.

Private Slots

Event Handling Slots

Slots for handling UI events.

These slots are connected to button clicks within the dialog.

- void onSubmit ()

  Handles the submission of a new testimonial.
- void onCancel ()

  Handles the cancellation of the testimonial submission.

Private Attributes

UI Elements

Private member variables representing UI components.

These members store pointers to the various input fields and controls used within the testimonial dialog.

- QLineEdit * m_authorEdit
  Input field for the author's name.
- QTextEdit * m_contentEdit
  Input field for the testimonial content.
- QCheckBox * m_doNotShowAgain
  Checkbox to control future display of the prompt.

8.23.1 Detailed Description

The TestimonialDialog class provides a user interface for submitting testimonials.

This dialog includes fields for the author's name, the testimonial content, and an option to prevent future prompts. It handles the submission and cancellation of testimonials.

8.23.2 Constructor & Destructor Documentation

8.23.2.1 TestimonialDialog()

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

Constructs a TestimonialDialog.

Parameters

parent	The parent widget of this dialog.

Here is the call graph for this function:

8.23.3 Member Function Documentation

8.23.3.1 onCancel

```
void TestimonialDialog::onCancel () [private], [slot]
```

Handles the cancellation of the testimonial submission.

Here is the call graph for this function:

Here is the caller graph for this function:

8.23.3.2 onSubmit

```
void TestimonialDialog::onSubmit () [private], [slot]
```

Handles the submission of a new testimonial.

Here is the call graph for this function:

Here is the caller graph for this function:

8.23.4 Member Data Documentation

8.23.4.1 m_authorEdit

```
QLineEdit* TestimonialDialog::m_authorEdit  [private]
```

Input field for the author's name.

8.23.4.2 m_contentEdit

```
QTextEdit* TestimonialDialog::m_contentEdit  [private]
```

Input field for the testimonial content.

8.23.4.3 m_doNotShowAgain

```
QCheckBox* TestimonialDialog::m_doNotShowAgain  [private]
```

Checkbox to control future display of the prompt.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.cpp

8.24 TestimonialManager Class Reference

Manages user testimonials, including storage, retrieval, and prompting logic.

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

Inheritance diagram for TestimonialManager:

Collaboration diagram for TestimonialManager:

Signals

User Interaction Signals

Signal emitted when conditions are met to prompt the user for a testimonial.

This signal is typically connected to a UI element that will display the prompt.

- void shouldPromptForTestimonial ()

  Emitted to prompt the user for a testimonial.

Public Member Functions

Core Testimonial Operations

Core testimonial management functions.

These methods allow adding new testimonials, retrieving satisfactory ones for display, and checking if a specific user has already submitted a testimonial.

- void addTestimonial (const Testimonial &testimonial)
  Adds a new testimonial to the manager and persists it.
- QVector< Testimonial > getSatisfactoryTestimonials () const
  Returns a vector of testimonials marked as satisfactory.
- bool hasUserGivenTestimonial (const QString &username) const
  Checks if a user with the given username has already submitted a testimonial.

User Prompt Preferences

User preference management for testimonial prompts.

These methods handle the "do not show again" preference for testimonial prompts on a per-user basis.

- void setDoNotShowAgain (const QString &username, bool value)
  Sets the "do not show again" preference for a user.
- bool getDoNotShowAgain (const QString &username) const
  Retrieves the "do not show again" preference for a user.

Activity Time Tracking

Time tracking control for testimonial prompts.

These methods start and stop the timer that tracks user activity, which is used to determine when to prompt for a testimonial.

- void startTrackingTime ()
  Starts the timer for tracking user activity time.
- void stopTrackingTime ()
  Stops the timer for tracking user activity time.

Static Public Member Functions

Singleton Access

Provides access to the singleton instance of TestimonialManager.

Ensures that only one instance of TestimonialManager exists throughout the application.

- static TestimonialManager & getInstance ()
  Returns a reference to the singleton TestimonialManager instance.

Private Slots

API Response Slots

API client response handlers.

These private slots are connected to signals from the ApiClient to process the results of network requests related to testimonials.

- void onGoodGetResponse (const QString &json)
  Slot called when a GET request for testimonials succeeds.
- void onBadGetResponse (const QString &error)
  Slot called when a GET request for testimonials fails.
- void onGoodPostResponse ()
  Slot called when a POST request to save testimonials succeeds.
- void onBadPostResponse (const QString &error)
  Slot called when a POST request to save testimonials fails.

Private Member Functions

Singleton Constructor/Destructor

Private constructor and destructor for the singleton pattern.

Ensures that TestimonialManager cannot be instantiated directly from outside the class.

- TestimonialManager ()
  Private constructor to enforce singleton pattern. Initializes API connections and timer.
- ~TestimonialManager ()
  Private destructor. Saves testimonials before destruction.

Internal Logic

Internal data persistence and prompt logic.

These methods handle the loading and saving of testimonial data (now via API) and the logic for checking if a testimonial prompt should be displayed.

- void loadTestimonials ()
  Loads testimonials (now primarily via API client in constructor).
- void saveTestimonials ()
  Saves all current testimonials (via API client).
- void checkTimeAndPrompt ()
  Checks accumulated user time and emits `shouldPromptForTestimonial` if criteria are met.

Member Data

Private member variables for TestimonialManager.

These members store testimonials, manage timing, user preferences, and utility objects.

- QVector< Testimonial > m_testimonials
  In-memory storage for all loaded testimonials.
- QTimer * m_trackingTimer
  Timer used to periodically check if a testimonial prompt is needed.
- QHash< QString, bool > m_doNotShowAgain
  Stores user preferences for not showing the testimonial prompt again. Key is username.
- QHash< QString, int > m_userTimeTracking
  Tracks accumulated active time for each user in minutes. Key is username.
- ApiClient client
  API client instance for network communication.
- Parser parse
  Parser instance for JSON serialization/deserialization.
- static const int INITIAL_PROMPT_TIME = 30 * 60
  Time in seconds before the first testimonial prompt (30 minutes).
- static const int REPEAT_PROMPT_TIME = 60 * 60
  Time in seconds for subsequent testimonial prompts if not dismissed (1 hour).

8.24.1 Detailed Description

Manages user testimonials, including storage, retrieval, and prompting logic.

This class is a singleton that handles all operations related to testimonials. It interacts with an ApiClient to load and save testimonials from/to a webservice, tracks user activity time to determine when to prompt for a testimonial, and manages user preferences regarding these prompts.

8.24.2 Constructor & Destructor Documentation

8.24.2.1 TestimonialManager()

```
TestimonialManager::TestimonialManager () [private]
```

Private constructor to enforce singleton pattern. Initializes API connections and timer.

Here is the call graph for this function:

Here is the caller graph for this function:

8.24.2.2 ~TestimonialManager()

```
TestimonialManager::~TestimonialManager () [private]
```

Private destructor. Saves testimonials before destruction.

Here is the call graph for this function:

8.24.3 Member Function Documentation

8.24.3.1 addTestimonial()

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

Adds a new testimonial to the manager and persists it.

Here is the call graph for this function:

Here is the caller graph for this function:

8.24.3.2 checkTimeAndPrompt()

```
void TestimonialManager::checkTimeAndPrompt () [private]
```

Checks accumulated user time and emits `shouldPromptForTestimonial` if criteria are met.

Here is the call graph for this function:

Here is the caller graph for this function:

8.24.3.3 getDoNotShowAgain()

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

Retrieves the "do not show again" preference for a user.

Here is the caller graph for this function:

8.24.3.4 getInstance()

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

Returns a reference to the singleton TestimonialManager instance.

Here is the call graph for this function:

Here is the caller graph for this function:

8.24.3.5 getSatisfactoryTestimonials()

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

Returns a vector of testimonials marked as satisfactory.

Here is the caller graph for this function:

8.24.3.6 hasUserGivenTestimonial()

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

Checks if a user with the given username has already submitted a testimonial.

Here is the caller graph for this function:

8.24.3.7 loadTestimonials()

```
void TestimonialManager::loadTestimonials () [private]
```

Loads testimonials (now primarily via API client in constructor).

8.24.3.8 onBadGetResponse

```
void TestimonialManager::onBadGetResponse (
            const QString & error) [private], [slot]
```

Slot called when a GET request for testimonials fails.

Here is the caller graph for this function:

8.24.3.9 onBadPostResponse

```
void TestimonialManager::onBadPostResponse (
            const QString & error)  [private], [slot]
```

Slot called when a POST request to save testimonials fails.

Here is the caller graph for this function:

8.24.3.10 onGoodGetResponse

```
void TestimonialManager::onGoodGetResponse (
            const QString & json)  [private], [slot]
```

Slot called when a GET request for testimonials succeeds.

Here is the caller graph for this function:

8.24.3.11 onGoodPostResponse

```
void TestimonialManager::onGoodPostResponse ()  [private], [slot]
```

Slot called when a POST request to save testimonials succeeds.

Here is the caller graph for this function:

8.24.3.12 saveTestimonials()

```
void TestimonialManager::saveTestimonials () [private]
```

Saves all current testimonials (via API client).

Here is the call graph for this function:

Here is the caller graph for this function:

8.24.3.13 setDoNotShowAgain()

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

Sets the "do not show again" preference for a user.

Here is the call graph for this function:

Here is the caller graph for this function:

8.24.3.14 shouldPromptForTestimonial

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

Emitted to prompt the user for a testimonial.

Here is the caller graph for this function:

8.24.3.15 startTrackingTime()

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

Starts the timer for tracking user activity time.

Here is the caller graph for this function:

8.24.3.16 stopTrackingTime()

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

Stops the timer for tracking user activity time.

8.24.4 Member Data Documentation

8.24.4.1 client

```
ApiClient TestimonialManager::client [private]
```

API client instance for network communication.

8.24.4.2 INITIAL_PROMPT_TIME

```
const int TestimonialManager::INITIAL_PROMPT_TIME = 30 * 60 [static], [private]
```

Time in seconds before the first testimonial prompt (30 minutes).

8.24.4.3 m_doNotShowAgain

```
QHash<QString, bool> TestimonialManager::m_doNotShowAgain [private]
```

Stores user preferences for not showing the testimonial prompt again. Key is username.

8.24.4.4 m_testimonials

```
QVector<Testimonial> TestimonialManager::m_testimonials [private]
```

In-memory storage for all loaded testimonials.

8.24.4.5 m_trackingTimer

```
QTimer* TestimonialManager::m_trackingTimer [private]
```

Timer used to periodically check if a testimonial prompt is needed.

8.24.4.6 m_userTimeTracking

```
QHash<QString, int> TestimonialManager::m_userTimeTracking [private]
```

Tracks accumulated active time for each user in minutes. Key is username.

8.24.4.7 parse

```
Parser TestimonialManager::parse [private]
```

Parser instance for JSON serialization/deserialization.

8.24.4.8 REPEAT_PROMPT_TIME

```
const int TestimonialManager::REPEAT_PROMPT_TIME = 60 * 60  [static], [private]
```

Time in seconds for subsequent testimonial prompts if not dismissed (1 hour).

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.cpp

8.25 TestimonialsDisplayDialog Class Reference

The TestimonialsDisplayDialog class provides a UI to display customer testimonials.

```
#include <TestimonialsDisplayDialog.h>
```

Inheritance diagram for TestimonialsDisplayDialog:

Collaboration diagram for TestimonialsDisplayDialog:

Public Member Functions

- TestimonialsDisplayDialog (QWidget ∗parent=nullptr)

  Constructs a TestimonialsDisplayDialog.

Private Member Functions

- void refreshTestimonials ()

  Refreshes the displayed list of testimonials.

Private Attributes

- QVBoxLayout ∗ m_testimonialsLayout

  Private member variables representing UI components.

8.25.1 Detailed Description

The TestimonialsDisplayDialog class provides a UI to display customer testimonials.

This dialog fetches and displays testimonials from the TestimonialManager. It includes a button to open the TestimonialDialog for new submissions and dynamically updates the displayed list.

8.25.2 Constructor & Destructor Documentation

8.25.2.1 TestimonialsDisplayDialog()

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

Constructs a TestimonialsDisplayDialog.

Parameters

parent	The parent widget of this dialog.

Here is the call graph for this function:

8.25.3 Member Function Documentation

8.25.3.1 refreshTestimonials()

```
void TestimonialsDisplayDialog::refreshTestimonials () [private]
```

Refreshes the displayed list of testimonials.

Clears the current list and repopulates it with testimonials retrieved from the TestimonialManager. Here is the call graph for this function:

Here is the caller graph for this function:

8.25.4 Member Data Documentation

8.25.4.1 m_testimonialsLayout

```
QVBoxLayout* TestimonialsDisplayDialog::m_testimonialsLayout  [private]
```

Private member variables representing UI components.

These members store pointers to the layout managing the display of testimonials. Layout for arranging individual testimonial widgets.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialsDisplayDialog.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialsDisplayDialog.cpp

8.26 Text Class Reference

The Text class.

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

Inheritance diagram for Text:

Collaboration diagram for Text:

Public Member Functions

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

  Text.
- void Draw (QWidget ∗renderArea) override

  Draw - Draws the text to the assigned coords.

- double Perimeter () const override

  Perimeter - Returns the perimater of the text box.
- double Area () const override

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

  isPointInside - Returns True if point is inside the text box
- int getLength () const

  Accessor Functions.
- int getWidth () const
- QString getTextString () const
- GlobalColor getTextColor () const
- QFont getFont () const
- AlignmentFlag getTextAlignment () const
- int getFontStyle () const
- QFont::Weight getFontWeight () const
- void setText (QString text)

  Mutator Functions.
- void setLength (int newLength)
- void setWidth (int newWidth)
- void setX (int newX)
- void setY (int newY)
- void setAlignment (Qt::AlignmentFlag alignment)
- QFont & setInternalFont ()

Public Member Functions inherited from Shape

- Shape (int shapeId, 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 setShapeType (string shapeType)

  Accessor Functions.
- 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 ()

Private Attributes

- int length

  length of text box
- int width

  width of text box
- QString textString

  string of text being displayed
- GlobalColor textColor

  text color
- QFont font

  text font
- AlignmentFlag textAlignment

  text alignment

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*

8.26.1 Detailed Description

The Text class.

8.26.2 Constructor & Destructor Documentation

8.26.2.1 Text()

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

Text.

Parameters

shapeType	- Used in Shape constructor MIL
coords	- Used in Shape constructor MIL
textString	- String of text being displayed
textColor	- Color of text
textAlignment	- Alignment of text
font	- Text font
length	- Length of text box
width	- Width of text box

Here is the call graph for this function:

8.26.3 Member Function Documentation

8.26.3.1 Area()

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

Area - Returns the area of the text box.

Returns

Implements Shape.

8.26.3.2 Draw()

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

Draw - Draws the text to the assigned coords.

Parameters

renderArea	- The renderArea being drawn on

Implements Shape.

Here is the call graph for this function:

8.26.3.3 getFont()

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

Here is the caller graph for this function:

8.26.3.4 getFontStyle()

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

Here is the caller graph for this function:

8.26.3.5 getFontWeight()

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

Here is the caller graph for this function:

8.26.3.6 getLength()

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

Accessor Functions.

Here is the caller graph for this function:

8.26.3.7 getTextAlignment()

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

Here is the caller graph for this function:

8.26.3.8 getTextColor()

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

Here is the caller graph for this function:

8.26.3.9 getTextString()

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

Here is the caller graph for this function:

8.26.3.10 getWidth()

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

Here is the caller graph for this function:

8.26.3.11 isPointInside()

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

isPointInside - Returns True if point is inside the text box

Parameters

point	- point being read

Returns

Implements Shape.

Here is the call graph for this function:

8.26.3.12 Perimeter()

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

Perimeter - Returns the perimater of the text box.

Returns

Implements Shape.

8.26.3.13 setAlignment()

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

Here is the caller graph for this function:

8.26.3.14 setInternalFont()

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

Here is the caller graph for this function:

8.26.3.15 setLength()

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

Here is the caller graph for this function:

8.26.3.16 setText()

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

Mutator Functions.

Here is the caller graph for this function:

8.26.3.17 setWidth()

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

Here is the caller graph for this function:

8.26.3.18 setX()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.26.3.19 setY()

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

Implements Shape.

Here is the call graph for this function:

Here is the caller graph for this function:

8.26.4 Member Data Documentation

8.26.4.1 font

```
QFont Text::font [private]
```

text font

8.26.4.2 length

```
int Text::length [private]
```

length of text box

8.26.4.3 textAlignment

```
AlignmentFlag Text::textAlignment [private]
```

text alignment

8.26.4.4 textColor

```
GlobalColor Text::textColor [private]
```

text color

8.26.4.5 textString

```
QString Text::textString [private]
```

string of text being displayed

8.26.4.6 width

```
int Text::width [private]
```

width of text box

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.cpp

8.27 UserAccount Class Reference

Represents a user account within the application.

```
#include <UserAccount.h>
```

Collaboration diagram for UserAccount:

Public Member Functions

Constructors and Destructor

Manages the lifecycle of UserAccount objects.

These methods handle the creation, initialization, and destruction of UserAccount objects.

- UserAccount ()

Default constructor. Initializes as a guest user.

- UserAccount (QString username, QString password, bool admin)

  Parameterized constructor. Initializes with specified credentials and admin status.
- ~UserAccount ()

  Destructor.

Copy and Move Semantics

Defines how UserAccount objects are copied and moved.

These methods define how UserAccount objects are copied and moved, ensuring proper resource management.

- UserAccount (const UserAccount &other)

  Copy constructor.
- UserAccount & operator= (const UserAccount &other)

  Copy assignment operator.
- UserAccount (UserAccount &&other) noexcept

  Move constructor.
- UserAccount & operator= (UserAccount &&other) noexcept

  Move assignment operator.

Getters

Public getters for accessing user account data.

These methods provide read-only access to the private member variables of the UserAccount object.

- QString getUsername () const

  Returns the username of the account.
- QString getPassword () const

  Returns the password of the account.
- bool isAdmin () const

  Returns true if the user has administrative privileges, false otherwise.

Setters

Public setters for modifying user account data.

These methods allow modification of the username, password, and administrative status of the UserAccount object.

- void setUsername (const QString &username)

  Sets the username for the account.
- void setPassword (const QString &password)

  Sets the password for the account.
- void setAdmin (bool admin)

  Sets the administrative status for the account.
- void setUserAccount (const QString &username, const QString &password, bool admin)

  Sets all properties of the user account at once.

Private Attributes

Member Data

Private member variables storing user account attributes.

These variables store the core information for each user account.

- QString username

  The username associated with the account.
- QString password

  The password for the account. For guest users, this might be NULL or empty.
- bool admin

  Flag indicating whether the user has administrative privileges.

8.27.1 Detailed Description

Represents a user account within the application.

This class encapsulates user credentials (username and password) and administrative status. It provides constructors, destructors, copy/move semantics, and getters/setters for its properties.

8.27.2 Constructor & Destructor Documentation

8.27.2.1 UserAccount() [1/4]

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

Default constructor. Initializes as a guest user.

Here is the caller graph for this function:

8.27.2.2 UserAccount() [2/4]

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

Parameterized constructor. Initializes with specified credentials and admin status.

8.27.2.3 ~UserAccount()

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

Destructor.

8.27.2.4 UserAccount() [3/4]

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

Copy constructor.

Here is the call graph for this function:

8.27.2.5 UserAccount() [4/4]

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

Move constructor.

Here is the call graph for this function:

8.27.3 Member Function Documentation

8.27.3.1 getPassword()

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

Returns the password of the account.

Here is the caller graph for this function:

8.27.3.2 getUsername()

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

Returns the username of the account.

Here is the caller graph for this function:

8.27.3.3 isAdmin()

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

Returns true if the user has administrative privileges, false otherwise.

Here is the caller graph for this function:

8.27.3.4 operator=() [1/2]

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

Copy assignment operator.

Here is the call graph for this function:

8.27.3.5 operator=() [2/2]

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

Move assignment operator.

Here is the call graph for this function:

8.27.3.6 setAdmin()

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

Sets the administrative status for the account.

8.27.3.7 setPassword()

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

Sets the password for the account.

8.27.3.8 setUserAccount()

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

Sets all properties of the user account at once.

8.27.3.9 setUsername()

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

Sets the username for the account.

8.27.4 Member Data Documentation

8.27.4.1 admin

```
bool UserAccount::admin [private]
```

Flag indicating whether the user has administrative privileges.

8.27.4.2 password

```
QString UserAccount::password [private]
```

The password for the account. For guest users, this might be NULL or empty.

8.27.4.3 username

```
QString UserAccount::username [private]
```

The username associated with the account.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.cpp

8.28 UserManager Class Reference

Manages user accounts, including creation, authentication, and persistence.

```
#include <UserManager.h>
```

Inheritance diagram for UserManager:

Collaboration diagram for UserManager:

Signals

User Status Signals

Signals for UI interaction and status updates.

These signals are emitted to notify the frontend about changes in user data, authentication status, or general operational messages.

- void userChanged ()

  Emitted when the list of users changes (add, modify, delete).
- void userNotChanged (const QString &message)

  Emitted when an operation that would change users fails (e.g., user not found, username taken).
- void statusMessage (const QString &message)

  Emitted to provide general status updates or error messages.
- void userAuthenticated (const UserAccount *currUser)

  Emitted upon successful user authentication, providing the authenticated user.
- void authenticationFailed (const QString &message)

  Emitted when user authentication fails.

Public Member Functions

Constructor and Destructor

Manages the lifecycle of UserManager objects.

Handles the initialization and cleanup of the UserManager. The constructor sets up connections to the ApiClient for handling responses to user data requests and loads initial user data.

- UserManager (QObject *parent=nullptr)

  Constructor. Initializes ApiClient connections and loads users.
- ~UserManager ()

  Destructor. Cleans up dynamically allocated UserAccount objects.

Current User Access

Accessor for the current user.

Provides a reference to the currently authenticated user account. If no user is authenticated, this typically points to a guest account.

- UserAccount * getCurrUserRef ()

  Returns a pointer to the currently authenticated UserAccount object.

User Account Management

Core functions for managing user accounts.

These methods provide functionalities to add, modify, delete, load, save, and authenticate user accounts. Most operations will trigger signals to update the UI or indicate status.

- void addUser (const QString username, const QString password, const bool admin)
- void modifyUser (const QString username, const QString password, const bool admin)
- void deleteUser (QString username)

  Deletes a specific user account by username.
- void deleteAllUsers ()

  Deletes all user accounts from the backend.
- void loadUsers ()

  Initiates loading of all user accounts from the webservice.
- void saveUsers ()

  Saves all current user accounts to the webservice.
- void authenticate (const QString username, const QString password)

Private Slots

API Response Slots

API client response handlers.

These private slots are connected to signals from the ApiClient to process the results of network requests related to user data.

- void onGoodGetResponse (const QString &json)
  Slot called when a GET request for user data succeeds.
- void onBadGetResponse (const QString &errorMsg)
  Slot called when a GET request for user data fails.
- void onGoodPostResponse ()
  Slot called when a POST request to save user data succeeds.
- void onBadPostResponse (const QString &errorMsg)
  Slot called when a POST request to save user data fails.
- void onGoodDeleteResponse ()
  Slot called when a DELETE request for user data succeeds.
- void onBadDeleteResponse (const QString &errorMsg)
  Slot called when a DELETE request for user data fails.

Private Attributes

Member Data

Private member variables for UserManager.

These members store the current user, all known users, and utility objects.

- UserAccount * currUser
  Pointer to the currently authenticated user. Defaults to a guest user.
- alpha::vector< UserAccount * > users
  Vector storing pointers to all loaded UserAccount objects.
- ApiClient client
  API client instance for network communication with the user data backend.
- Parser parse
  Parser instance for JSON serialization/deserialization of user data.

8.28.1 Detailed Description

Manages user accounts, including creation, authentication, and persistence.

This class handles all operations related to user accounts, such as adding, modifying, deleting, and authenticating users. It interacts with an ApiClient to load and save user data from/to a webservice and uses a Parser for data serialization/deserialization.

8.28.2 Constructor & Destructor Documentation

8.28.2.1 UserManager()

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

Constructor. Initializes ApiClient connections and loads users.

Here is the call graph for this function:

8.28.2.2 ~UserManager()

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

Destructor. Cleans up dynamically allocated UserAccount objects.

8.28.3 Member Function Documentation

8.28.3.1 addUser()

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

Parameters

username	Adds a new user account and attempts to authenticate with the new credentials.

Here is the call graph for this function:

8.28.3.2 authenticate()

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

Parameters

username	Authenticates a user against the loaded user accounts.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.3 authenticationFailed

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

Emitted when user authentication fails.

Here is the caller graph for this function:

8.28.3.4 deleteAllUsers()

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

Deletes all user accounts from the backend.

8.28.3.5 deleteUser()

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

Deletes a specific user account by username.

Here is the call graph for this function:

8.28.3.6 getCurrUserRef()

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

Returns a pointer to the currently authenticated UserAccount object.

8.28.3.7 loadUsers()

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

Initiates loading of all user accounts from the webservice.

8.28.3.8 modifyUser()

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

Parameters

username	Modifies an existing user account's details (password, admin status).

Here is the call graph for this function:

8.28.3.9 onBadDeleteResponse

```
void UserManager::onBadDeleteResponse (
            const QString & errorMsg) [private], [slot]
```

Slot called when a DELETE request for user data fails.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.10 onBadGetResponse

```
void UserManager::onBadGetResponse (
            const QString & errorMsg)  [private], [slot]
```

Slot called when a GET request for user data fails.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.11 onBadPostResponse

```
void UserManager::onBadPostResponse (
            const QString & errorMsg)  [private], [slot]
```

Slot called when a POST request to save user data fails.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.12 onGoodDeleteResponse

```
void UserManager::onGoodDeleteResponse () [private], [slot]
```

Slot called when a DELETE request for user data succeeds.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.13 onGoodGetResponse

```
void UserManager::onGoodGetResponse (
            const QString & json)  [private], [slot]
```

Slot called when a GET request for user data succeeds.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.14 onGoodPostResponse

```
void UserManager::onGoodPostResponse ()  [private], [slot]
```

Slot called when a POST request to save user data succeeds.

Here is the call graph for this function:

Here is the caller graph for this function:

8.28.3.15 saveUsers()

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

Saves all current user accounts to the webservice.

Here is the caller graph for this function:

8.28.3.16 statusMessage

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

Emitted to provide general status updates or error messages.

Here is the caller graph for this function:

8.28.3.17 userAuthenticated

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

Emitted upon successful user authentication, providing the authenticated user.

Here is the caller graph for this function:

8.28.3.18 userChanged

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

Emitted when the list of users changes (add, modify, delete).

8.28.3.19 userNotChanged

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

Emitted when an operation that would change users fails (e.g., user not found, username taken).

Here is the caller graph for this function:

8.28.4 Member Data Documentation

8.28.4.1 client

```
ApiClient UserManager::client  [private]
```

API client instance for network communication with the user data backend.

8.28.4.2 currUser

```
UserAccount* UserManager::currUser [private]
```

Pointer to the currently authenticated user. Defaults to a guest user.

8.28.4.3 parse

```
Parser UserManager::parse [private]
```

Parser instance for JSON serialization/deserialization of user data.

8.28.4.4 users

```
alpha::vector<UserAccount*> UserManager::users [private]
```

Vector storing pointers to all loaded UserAccount objects.

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.h
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_UserManager.cpp
- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.cpp

8.29 alpha::vector< T > Class Template Reference

```
#include <vector.h>
```

Inheritance diagram for alpha::vector< T >:

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)

Private Attributes

- int size_v
- T * elem
- int space

8.29.1 Member Typedef Documentation

8.29.1.1 const_iterator

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

8.29.1.2 iterator

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

8.29.2 Constructor & Destructor Documentation

8.29.2.1 vector() [1/4]

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

Here is the caller graph for this function:

8.29.2.2 vector() [2/4]

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

8.29.2.3 vector() [3/4]

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

Here is the call graph for this function:

8.29.2.4 vector() [4/4]

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

Here is the call graph for this function:

8.29.2.5 ~vector()

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

8.29.3 Member Function Documentation

8.29.3.1 begin() [1/2]

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

8.29.3.2 begin() [2/2]

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

8.29.3.3 capacity()

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

8.29.3.4 end() [1/2]

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

Here is the caller graph for this function:

8.29.3.5 end() [2/2]

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

8.29.3.6 erase()

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

Here is the call graph for this function:

8.29.3.7 insert()

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

Here is the call graph for this function:

8.29.3.8 operator=() [1/2]

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

Here is the call graph for this function:

8.29.3.9 operator=() [2/2]

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

Here is the call graph for this function:

8.29.3.10 operator[]() [1/2]

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

8.29.3.11 operator[]() [2/2]

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

8.29.3.12 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:

8.29.3.13 reserve()

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

Here is the caller graph for this function:

8.29.3.14 resize()

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

Here is the call graph for this function:

8.29.3.15 size()

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

Here is the caller graph for this function:

8.29.4 Member Data Documentation

8.29.4.1 elem

```
template<class T>
T* alpha::vector< T >::elem [private]
```

8.29.4.2 size_v

```
template<class T>
int alpha::vector< T >::size_v [private]
```

8.29.4.3 space

```
template<class T>
int alpha::vector< T >::space [private]
```

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

- /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/vector.h

Chapter 9

File Documentation

9.1 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.cpp File Reference

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

Include dependency graph for ApiClient.cpp:

9.2 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/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.

9.2.1 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()

9.3 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
00030 class ApiClient : public QObject {
00031     Q_OBJECT    //necessary macro for qt's compiler
00032
00033 public:
```

```
    explicit ApiClient(QObject* parent = nullptr);


    void GetShapes();
    void PostShapes(std::string json);
    void DeleteShapesAll();


    void GetRenderArea();
    void PostRenderArea(std::string json);
    void DeleteRenderAreaAll();


    void GetUsers();
    void PostUsers(std::string json);
    void DeleteUsersAll();


    void GetTestimonials();
    void PostTestimonials(std::string json);
    void DeleteTestimonialsAll();


signals:
    void GoodGetReply(const QString& json);
    void BadGetReply(const QString& error);
    void GoodPostReply();
    void BadPostReply(const QString& error);
    void GoodDeleteReply();
    void BadDeleteReply(const QString& error);


private slots:
    void AnalyzeGetReply();
    void AnalyzePostReply();
    void AnalyzeDeleteReply();


private:
    QNetworkAccessManager* manager;
};

#endif //API_CLIENT_H
```

9.4 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.cpp File Reference

`#include "AppDriver.h"`

Include dependency graph for AppDriver.cpp:

9.5 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.h File Reference

```
#include <QCoreApplication>
#include <QApplication>
#include <QObject>
#include <QTimer>
#include <QDebug>
#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

  Orchestrates the main application logic and connections.

9.6 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 <QTimer>
00007 #include <QDebug>
00008 #include "RenderAreaManager.h"
00009 #include "UserManager.h"
00010 #include "../frontend/mainwindow.h"
00011 #include "../frontend/renderarea.h"
00012
00022 class AppDriver : public QObject {
00023     Q_OBJECT
00024 public:
00032     AppDriver(QObject* parent = nullptr);
00033     ~AppDriver();
00034     void run();
00035     void shutdown();
00036     void loadAllData();
00038
00039
00040 private slots:
00049     void onRenderShapeAdded(Shape* shape);
00050     void onRenderShapeChanged(Shape* shape,
```

```
                                QString key,
                                int value);
    void onRenderShapeDeleted(const int trackerId);
    void onRenderDeleteAllShapes();

    void onNewUser(const QString username,
                   const QString password,
                   const bool admin);
    void onUserModified(const QString username,
                        const QString password,
                        const bool admin);
    void onUserDeleted(const QString username);
    void onDeleteAllUsers();
    void onLoginAttempt(const QString username,
                        const QString password);


private:
    MainWindow* mainWindow;
    RenderAreaManager* renderedShapes;
    UserManager* user;


    void connectFrontendToDriver();
    void connectManagersToFrontend();
};

#endif // APP_DRIVER_H
```

9.7 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/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

Variables

- static Q_CONSTINIT const uint qt_meta_data_ZN9ApiClientE [ ]

9.7.1 Macro Definition Documentation

9.7.1.1 Q_CONSTINIT

```
#define Q_CONSTINIT
```

9.7.2 Variable Documentation

9.7.2.1 qt_meta_data_ZN9ApiClientE

```
Q_CONSTINIT const uint qt_meta_data_ZN9ApiClientE[]  [static]
```

9.8 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-↩ Modeler/src/backend/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

Variables

- static Q_CONSTINIT const uint qt_meta_data_ZN11UserManagerE [ ]

9.8.1 Macro Definition Documentation

9.8.1.1 Q_CONSTINIT

```
#define Q_CONSTINIT
```

9.8.2 Variable Documentation

9.8.2.1 qt_meta_data_ZN11UserManagerE

```
Q_CONSTINIT const uint qt_meta_data_ZN11UserManagerE[]  [static]
```

9.9 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.cpp File Reference

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

Include dependency graph for Parser.cpp:

9.10 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h File Reference

```
#include <iostream>
#include <cctype>
#include <QVector>
#include <QJsonDocument>
#include <QJsonArray>
#include <QJsonObject>
#include <string>
#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

  Provides functionality for parsing JSON data to C++ objects and vice-versa.
- struct Parser::MorphicShape

  Internal accumulator structure for parsing shape data from JSON.
- struct Parser::RawUser

  Internal accumulator structure for parsing user account data from JSON.

9.11 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 <string>
00011 #include "../objects/vector.h"
00012 #include "../objects/all_shapes.h"
00013 #include "UserAccount.h"
00014 #include "Testimonial.h"
00015
00026 class Parser {
00027 public:
00029     Parser() = default;
00031     ~Parser() = default;
00032
00033     //Disable copying and moving
00035     Parser(const Parser&) = delete;
00037     Parser& operator=(const Parser&) = delete;
```

```
    Parser(Parser&&) = delete;
    Parser& operator=(Parser&&) = delete;

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

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

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

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

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

    static QVector<Testimonial> JsonToTestimonials(const std::string& json);

    static std::string TestimonialsToJson(const QVector<Testimonial>& testimonials);

private:
    /*==================================== Forward Parser Subroutines ==========================================*/
    struct MorphicShape {
        std::string shapeType = "";
        int shapeId = 0;
        int trackerId = 0;
        alpha::vector<int> shapeDimensions;
        QPen pen = QPen();
        QBrush brush = QBrush();
        QPoint coords = QPoint();

        QString textString;
        GlobalColor textColor;
        QFont font;
        AlignmentFlag textAlignment;
    };

    struct RawUser {
            QString username;
            QString password;
            bool    admin       = false;
            bool    hasUsername = false;
            bool    hasPassword = false;
            bool    hasAdmin    = false;
        };

    MorphicShape ParseJsonObject(const std::string json, size_t &index);

    void UpdateAccumulator(const std::string &key, const std::string &value, MorphicShape &tempShape);

    Shape* BuildShape(MorphicShape tempShape);

    void SkipWhitespace(const std::string& json, size_t& index);

    std::string ExtractKey(const std::string& json, size_t &index);

    std::string ExtractValue(const std::string& json, size_t &index);

    std::string ExtractInteger(const std::string& json, size_t &index);

    std::string ExtractArray(const std::string& json, size_t &index);

    std::string ExtractLiteral(const std::string& json, size_t& index);

    alpha::vector<int> StringToVector(const std::string &value);

    /*==================================== Reverse Parser Subroutines ==========================================*/

    std::string AppendCommonShapeData(const Shape* shape);

    std::string AppendBrushData(const Shape* shape);

    std::string AppendTextData(const Shape* shape);

    std::string GetShapeDimensions(const Shape* shape);

    std::string GetColor(const QColor &objectColor);

    std::string GetPenStyle(const Shape* shape);

    std::string GetPenCapStyle(const Shape* shape);

    std::string GetPenJoinStyle(const Shape* shape);

    std::string GetBrushStyle(const Shape* shape);

    std::string GetAlignmentFlag(const Text* text);

```

```
    std::string GetFontStyle(const Text* text);

    std::string GetFontWeight(const Text* text);

    static void   UpdateUserAccumulator(const std::string& key, const std::string& value, RawUser&
acc);
};

#endif // PARSER_H
```

9.12 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.cpp File Reference

```
#include "RenderAreaManager.h"
```

Include dependency graph for RenderAreaManager.cpp:

9.13 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.h File Reference

```
#include <QObject>
#include <QString>
#include <QDebug>
#include <QBrush>
#include <QPen>
#include "ApiClient.h"
#include "Parser.h"
#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

  Manages the shapes to be rendered and interacts with the backend API.

9.14 RenderAreaManager.h

Go to the documentation of this file.

```
#ifndef RENDER_AREA_MANAGER
#define RENDER_AREA_MANAGER

#include <QObject>
#include <QString>
#include <QDebug>
#include <QBrush>
#include <QPen>
#include "ApiClient.h"
#include "Parser.h"
#include <QString>
#include "../objects/all_shapes.h" // ShapeId enum located here
#include "../objects/vector.h"

class RenderAreaManager : public QObject {
    Q_OBJECT

public:
    explicit RenderAreaManager(QObject* parent = nullptr);
    ~RenderAreaManager();

    alpha::vector<Shape*>* getShapesRef();

    void addShape(Shape* shape);
    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();


signals:
    void renderAreaChanged();
    void renderAreaNotChanged(const QString &message);
    void statusMessage(const QString &message);

```

```
private slots:
    void onGoodGetResponse(const QString &json);
    void onBadGetResponse(const QString &errorMsg);
    void onGoodPostResponse();
    void onBadPostResponse(const QString &errorMsg);
    void onGoodDeleteResponse();
    void onBadDeleteResponse(const QString &errorMsg);


private:
    alpha::vector<Shape*> renderedShapes;
    ApiClient client;
    Parser parse;
};

#endif // RENDER_AREA_MANAGER
```

9.15 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.cpp File Reference

```
#include "ShapesManager.h"
```

Include dependency graph for ShapesManager.cpp:

9.16 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.h File Reference

```
#include <QObject>
#include <QString>
#include <QDebug>
#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

9.16.1 Detailed Description

Note

This class is deprecated and no longer used.

9.17 ShapesManager.h

Go to the documentation of this file.

```
00001
00006
00007 #ifndef SHAPES_MANAGER
00008 #define SHAPES_MANAGER
00009
00010 #include <QObject>
00011 #include <QString>
00012 #include <QDebug>
00013 #include "ApiClient.h"
00014 #include "Parser.h"
00015 #include "../objects/all_shapes.h"
00016 #include "../objects/vector.h"
00017
00018 class ShapesManager : public QObject {
00019     Q_OBJECT
00020 public:
00029     explicit ShapesManager(QObject* parent = nullptr);
00030     ~ShapesManager();
00031
00035     alpha::vector<Shape*>* getShapesRef();
00036
00045     void addShape(Shape* shape);
00046     void modifyShape(Shape* shape);
00047     void deleteShape(const int trackerId);
00048     void deleteAllShapes();
00049     void loadShapes();
```

```
    void saveShapes();



signals:
    void shapesChanged();
    void shapesNotChanged(const QString &messsage);
    void statusMessage(const QString &message);


private slots:
    void onGoodGetResponse(const QString &json);
    void onBadGetResponse(const QString &errorMsg);
    void onGoodPostResponse();
    void onBadPostResponse(const QString &errorMsg);
    void onGoodDeleteResponse();
    void onBadDeleteResponse(const QString &errorMsg);


private:
    alpha::vector<Shape*> shapes;
    ApiClient client;
    Parser parse;
};

#endif // SHAPES_MANAGER
```

9.18 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.cpp File Reference

`#include "Testimonial.h"`

Include dependency graph for Testimonial.cpp:

9.19 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.h File Reference

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

`#include <QJsonValue>`

Include dependency graph for Testimonial.h:

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

Classes

- class Testimonial

  Represents a user testimonial.

9.20 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 #include <QJsonArray>
00008 #include <QJsonValue>
00009
00017 class Testimonial {
00018 public:
00027     Testimonial(const QString& author = "", const QString& content = "", bool isGuest = true);
00029
00037     QString getAuthor() const;
00038     QString getContent() const;
00039     QDateTime getTimestamp() const;
00040     bool isGuest() const;
00041     bool isSatisfactory() const;
00043
```

```
    void setIsSatisfactory(bool value);

    QJsonObject toJson() const;
    static Testimonial fromJson(const QJsonObject& json);

private:
    QString m_author;
    QString m_content;
    QDateTime m_timestamp;
    bool m_isGuest;
    bool m_isSatisfactory;
};

#endif // TESTIMONIAL_H
```

9.21 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.cpp File Reference

```
#include "TestimonialManager.h"
```

Include dependency graph for TestimonialManager.cpp:

9.22 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.h File Reference

```
#include <QObject>
#include <QVector>
#include <QTimer>
#include <QJsonDocument>
#include <QJsonArray>
#include <QJsonObject>
#include <QDebug>
#include <QHash>
#include "Testimonial.h"
#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

  Manages user testimonials, including storage, retrieval, and prompting logic.

9.23 TestimonialManager.h

Go to the documentation of this file.

```
#ifndef TESTIMONIALMANAGER_H
#define TESTIMONIALMANAGER_H

#include <QObject>
#include <QVector>
#include <QTimer>
#include <QJsonDocument>
#include <QJsonArray>
#include <QJsonObject>
#include <QDebug>
#include <QHash>
#include "Testimonial.h"
#include "ApiClient.h"
#include "Parser.h"

class TestimonialManager : public QObject {
    Q_OBJECT
public:
    static TestimonialManager& getInstance();

    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();

signals:
    void shouldPromptForTestimonial();

private slots:
    void onGoodGetResponse(const QString& json);
    void onBadGetResponse(const QString& error);
    void onGoodPostResponse();
    void onBadPostResponse(const QString& error);

private:
    TestimonialManager();
    ~TestimonialManager();
```

```
    void loadTestimonials();
    void saveTestimonials();
    void checkTimeAndPrompt();

    QVector<Testimonial> m_testimonials;
    QTimer* m_trackingTimer;
    QHash<QString, bool> m_doNotShowAgain;
    QHash<QString, int> m_userTimeTracking;

    ApiClient client;
    Parser parse;

    // timing constants for testimonial prompts
    static const int INITIAL_PROMPT_TIME = 30 * 60;
    static const int REPEAT_PROMPT_TIME = 60 * 60;
};

#endif // TESTIMONIALMANAGER_H
```

9.24 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.cpp File Reference

```
#include "UserAccount.h"
```

Include dependency graph for UserAccount.cpp:

9.25 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.h File Reference

```
#include <QString>
#include <utility>
```

Include dependency graph for UserAccount.h:

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

Classes

- class UserAccount

  Represents a user account within the application.

9.26 UserAccount.h

Go to the documentation of this file.

```
#ifndef USERACCOUNT_H
#define USERACCOUNT_H

#include <QString>
#include <utility> // For std::move

class UserAccount {
public:
    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(const QString& username);
    void setPassword(const QString& password);
    void setAdmin(bool admin);
    void setUserAccount(const QString& username, const QString& password, bool admin);

private:
    QString username;
    QString password;
    bool admin;
};

#endif // USERACCOUNT_H
```

9.27 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.cpp File Reference

```
#include "UserManager.h"
```

Include dependency graph for UserManager.cpp:

9.28 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.h File Reference

```
#include <QObject>
#include <QString>
#include <QDebug>
#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

  Manages user accounts, including creation, authentication, and persistence.

9.29 UserManager.h

Go to the documentation of this file.

```
#ifndef USER_MANAGER_H
#define USER_MANAGER_H

#include <QObject>
#include <QString>
#include <QDebug>
#include <cctype>
#include <string>
#include "ApiClient.h"
#include "Parser.h"
#include "UserAccount.h"
#include "../objects/vector.h" // Assuming this is a custom vector implementation

class UserManager : public QObject {
    Q_OBJECT
public:
    explicit UserManager(QObject* parent = nullptr);
    ~UserManager();

    UserAccount* getCurrUserRef();

    void addUser(const QString username,
                 const QString password,
                 const bool admin);
    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);

signals:
    void userChanged();
    void userNotChanged(const QString &message);
    void statusMessage(const QString &message);
    void userAuthenticated(const UserAccount* currUser);
    void authenticationFailed(const QString &message);

private slots:
    void onGoodGetResponse(const QString &json);
    void onBadGetResponse(const QString &errorMsg);
    void onGoodPostResponse();
```

```
    void onBadPostResponse(const QString &errorMsg);
    void onGoodDeleteResponse();
    void onBadDeleteResponse(const QString &errorMsg);

private:
    UserAccount* currUser;
    alpha::vector<UserAccount*> users;
    ApiClient client;
    Parser parse;
};

#endif // USER_MANAGER_H
```

9.30 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/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

  The ColumnEditDelegate class.

9.31 ColumnEditDelegate.h

Go to the documentation of this file.

```
#ifndef COLUMNEDITDELEGATE_H
#define COLUMNEDITDELEGATE_H

#include <QStyledItemDelegate>

class ColumnEditDelegate : public QStyledItemDelegate
{
public:
    ColumnEditDelegate(QObject *parent = nullptr) : QStyledItemDelegate(parent) {}

    QWidget *createEditor(QWidget *parent,
                          const QStyleOptionViewItem &option,
                          const QModelIndex &index) const override
    {
        if (index.column() != 1 || !canEdit)
        {
            return nullptr;  // Only allows editing in column 1
        }

        return QStyledItemDelegate::createEditor(parent, option, index);
    }

    void setCanEdit(bool edit)
    {
        canEdit = edit;
    }

```

```
00047 private:
00048     bool canEdit = false;
00049 };
00050
00051 #endif // COLUMNEDITDELEGATE_H
```

9.32 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/darkstyle.qss File Reference

9.33 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Geoo.qss File Reference

9.34 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/lightstyle.qss File Reference

9.35 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.cpp File Reference

```
#include "loginwindow.h"
```

Include dependency graph for loginwindow.cpp:

9.36 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.h File Reference

```
#include <QDialog>
#include <QStackedWidget>
#include <QLineEdit>
#include <QPushButton>
#include <QVBoxLayout>
#include <QFormLayout>
#include <QHBoxLayout>
```

`#include <QLabel>`

Include dependency graph for loginwindow.h:

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

Classes

- class LoginWindow

  A dialog window for user login and signup.

9.37 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 #include <QVBoxLayout>
```

```
#include <QFormLayout>
#include <QHBoxLayout>
#include <QLabel>

class LoginWindow : public QDialog {
    Q_OBJECT

public:
    explicit LoginWindow(QWidget *parent = nullptr);

    QString username() const;
    QString password() const;

signals:
    void loginRequested(const QString &username, const QString &password);
    void signupRequested(const QString &username, const QString &password, const bool admin);

private slots:
    void showSignupPage();
    void showLoginPage();
    void attemptLogin();
    void attemptSignup();

private:
    QStackedWidget *stack;

    // --- login page widgets ---
    QWidget *loginPage;
    QLineEdit *loginUserEdit;
    QLineEdit *loginPassEdit;
    QPushButton *loginBtn;
    QPushButton *toSignupBtn;

    // --- signup page widgets ---
    QWidget *signupPage;
    QLineEdit *signupUserEdit;
    QLineEdit *signupPassEdit;
    QPushButton *signupBtn;
    QPushButton *backToLoginBtn;
};

#endif // LOGIN_WINDOW_H
```

9.38 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-↩ Modeler/src/backend/main.cpp File Reference

```
#include <QCoreApplication>
#include <QObject>
#include "ApiClient.h"
#include "Parser.h"
#include "UserManager.h"
#include "AppDriver.h"
```

Include dependency graph for main.cpp:

Functions

- void OnGoodGetResponseTest (const std::string &json)
- void OnBadGetResponseTest (const std::string &errorMsg)
- void OnGoodPostResponseTest ()
- void OnBadPostResponseTest (const std::string &errorMsg)

- ApiClient * GetConnectedClient ()
- std::string GetUsersTestString ()
- std::string GetShapeTestString ()
- std::string GetRenderAreaTestString ()
- int main (int argc, char *argv[ ])

Variables

- QCoreApplication * pApp = nullptr
- ApiClient * pClient = nullptr
- Parser parse

9.38.1 Function Documentation

9.38.1.1 GetConnectedClient()

```
ApiClient * GetConnectedClient ()
```

Here is the call graph for this function:

9.38.1.2 GetRenderAreaTestString()

```
std::string GetRenderAreaTestString ()
```

9.38.1.3 GetShapeTestString()

```
std::string GetShapeTestString ()
```

9.38.1.4 GetUsersTestString()

```
std::string GetUsersTestString ()
```

9.38.1.5 main()

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

Here is the call graph for this function:

9.38.1.6 OnBadGetResponseTest()

```
void OnBadGetResponseTest (
            const std::string & errorMsg)
```

Here is the caller graph for this function:

9.38.1.7 OnBadPostResponseTest()

```
void OnBadPostResponseTest (
            const std::string & errorMsg)
```

Here is the caller graph for this function:

9.38.1.8 OnGoodGetResponseTest()

```
void OnGoodGetResponseTest (
            const std::string & json)
```

Here is the caller graph for this function:

9.38.1.9 OnGoodPostResponseTest()

```
void OnGoodPostResponseTest ()
```

Here is the caller graph for this function:

9.38.2 Variable Documentation

9.38.2.1 pApp

```
QCoreApplication* pApp = nullptr
```

9.38.2.2 parse

```
Parser parse
```

9.38.2.3 pClient

```
ApiClient* pClient = nullptr
```

9.39 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/main.cpp File Reference

Main entry point for the 2D Graphics Modeler application.

```
#include "../backend/AppDriver.h"
#include <QApplication>
```

Include dependency graph for main.cpp:

Functions

- int main (int argc, char ∗argv[ ])

  The main function of the application.

9.39.1 Detailed Description

Main entry point for the 2D Graphics Modeler application.

Initializes the QApplication and the AppDriver, then starts the application event loop.

9.39.2 Function Documentation

9.39.2.1 main()

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

The main function of the application.

Parameters

argc	Number of command-line arguments.
argv	Array of command-line arguments.

Returns

Exit status of the application.

Here is the call graph for this function:

9.40 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-↩ Modeler/webservice-dockerized/main.cpp File Reference

```
#include <crow.h>
#include <crow/json.h>
#include <string>
#include <fstream>
#include <filesystem>
```

Include dependency graph for main.cpp:

Functions

- crow::json::rvalue get_json_file (const std::string &path)

  Reads and parses a JSON file.
- int main ()

9.40.1 Function Documentation

9.40.1.1 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:

9.40.1.2 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 "/webservice/database/shapes.json", parses it using Crow's JSON parser, and returns the data with the Content-Type header set to "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 "/webservice/database/shapes.↩ json". If the file cannot be opened, a 500 response is returned.

Clears all shapes.

Deletes the shapes.json file content by truncating the file, leaving it empty (an empty JSON array).

Retrieves render area data.

Reads JSON data from "/webservice/database/render_area.json", parses it using Crow's JSON parser, and returns the data with the Content-Type header set to "application/json".

Updates the render area data.

Accepts a POST request with JSON data and writes it to the file "/webservice/database/render_area.json". Returns a 500 response if the file cannot be opened.

Clears all render area data.

Deletes the render_area.json file content by truncating the file, leaving it empty (an empty JSON array).

Retrieves user account data.

Reads the JSON data from "/webservice/database/users.json", parses it, and returns it with Content-Type "application/json".

Updates users.json with new user data.

Accepts a POST request with JSON body and writes it to "/webservice/database/users.json".

Clears all user account data.

Deletes the users.json file content by truncating it, leaving an empty JSON array.

Retrieves testimonial data.

Reads the JSON array from "/webservice/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 "/webservice/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 at "/webservice/database", writing an empty JSON array. Returns 200 on success or 500 on failure.Here is the call graph for this function:

9.41 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.cpp File Reference

```
#include "mainwindow.h"
```

Include dependency graph for mainwindow.cpp:

9.42 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/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 <QAction>
#include <QMenuBar>
#include "ui_mainwindow.h"
#include "renderarea.h"
#include "loginwindow.h"
#include "TestimonialDialog.h"
#include "TestimonialsDisplayDialog.h"
#include "ColumnEditDelegate.h"
#include "../backend/UserAccount.h"
#include "../backend/TestimonialManager.h"
```

Include dependency graph for mainwindow.h:

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

Classes

- class MainWindow

  The main window of the 2D Graphics Modeler application.

Namespaces

- namespace Ui

9.43 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 <QStatusBar>
00016 #include <QVBoxLayout>
00017 #include <QPushButton>
00018 #include <QSpinBox>
00019 #include <QAction>
00020 #include <QMenuBar>
00021 #include "ui_mainwindow.h"
00022 #include "renderarea.h"
00023 #include "loginwindow.h"
00024 #include "TestimonialDialog.h"
00025 #include "TestimonialsDisplayDialog.h"
00026 #include "ColumnEditDelegate.h"
00027 #include "../backend/UserAccount.h"
00028 #include "../backend/TestimonialManager.h"
```

```
QT_BEGIN_NAMESPACE
namespace Ui {
class MainWindow;
}
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = nullptr);
    MainWindow(QWidget *parent,
               const alpha::vector<Shape*>* renderedShapes,
               const UserAccount* currUser);
    ~MainWindow();

    void drawShapes() const;
    void shapes_to_treeWidget();

signals:
    void shapeAdded(Shape* shape);
    void shapeChanged(Shape* shape, QString key, int value);
    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 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);

private slots:
    void onToggleStyle(bool checked = true);

    void on_actionnew_line_button_triggered();
    void on_actionnew_square_button_triggered();
    void on_actionnew_rectange_button_triggered();
    void on_actionnew_circle_button_triggered();
    void on_actionnew_ellipse_button_triggered();
    void on_actionnew_polyline_button_triggered();
    void on_actionnew_polygon_button_triggered();
    void on_actionnew_text_button_triggered();
    void on_actionremove_shape_button_triggered();

    void onSortMethodChanged(int index);

    void onTreeWidgetItemChanged(QTreeWidgetItem* item, int column);
    void onComboBoxChanged(int newIndex);
    void onSpinBoxChanged();

    void showTestimonialPrompt();
    void showTestimonialsDisplay();

    void onContactUsClicked();

private:
    Ui::MainWindow *ui;
    RenderArea *renderArea;
    const alpha::vector<Shape*>* renderShapes;
    const UserAccount* currUser;
    QLabel *userStatusLabel;
    ColumnEditDelegate* delegate;

    void addToShapeTree(Shape* shape);
    QString loadStyleSheet(const QString &path);

    QComboBox* createShapeTypeComboBox(const QString& currentShapeType);
    QSpinBox*  createPenWidthSpinBox(int currentPenWidth);
    QComboBox* createColorComboBox(int currentColor);
    QComboBox* createPenStyleComboBox(int currentPenStyle);
```

```
    QComboBox* createPenCapStyleComboBox(int currentPenCapStyle);
    QComboBox* createPenJoinStyleComboBox(int currentPenJoinStyle);
    QComboBox* createBrushStyleComboBox(int currentBrushStyle);
    QComboBox* createAlignmentComboBox(Qt::AlignmentFlag currentAlignment);
    QComboBox* createFontComboBox(QFont currentFont);
    QComboBox* createFontStyleComboBox(int currentFontStyle);
    QComboBox* createFontWeightComboBox(QFont::Weight currentFontWeight);

    void createShapeTableTab();
    QTabWidget* tabWidget;
    QTableWidget* shapeTable;
    QComboBox* sortDropdown;
    QComboBox* sortOrderDropdown;

    void selection_sort(alpha::vector<Shape*>& shapes,
                        bool (*compare)(const Shape*, const Shape*),
                        bool ascending);
    void populateShapeTable(const alpha::vector<Shape*>& shapes);
    static bool sortById(const Shape* a, const Shape* b);
    static bool sortByArea(const Shape* a, const Shape* b);
    static bool sortByPerimeter(const Shape* a, const Shape* b);

    void setupTestimonials();

    QWidget *contactUsWidget;
    QLabel *logoLabel;
    QLabel *teamNameLabel;
    QWidget *contactWindow;
};

#endif // MAINWINDOW_H
```

9.44 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.ui File Reference

9.45 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Medize.qss File Reference

9.46 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.cpp File Reference

`#include "renderarea.h"`

Include dependency graph for renderarea.cpp:

9.47 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.h File Reference

Defines the RenderArea class, a custom QWidget for drawing and interacting with 2D shapes.

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

Include dependency graph for renderarea.h:

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

Classes

- class RenderArea

  The RenderArea class provides a canvas for drawing 2D shapes.

9.47.1 Detailed Description

Defines the RenderArea class, a custom QWidget for drawing and interacting with 2D shapes.

9.48 renderarea.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include <QWidget>
00004 #include <QPainter>
00005 #include <QPen>
00006 #include <QMouseEvent>
00007
00008 #include "../objects/all_shapes.h" // Includes all shape types
00009 #include "../objects/vector.h"    // For alpha::vector
00010
00015
00022 class RenderArea : public QWidget
00023 {
00024     Q_OBJECT
00025
00026 public:
00031     explicit RenderArea(QWidget *parent = nullptr);
00032
00039     void mousePressEvent(QMouseEvent* event) override;
00040     void mouseMoveEvent(QMouseEvent* event) override;
00041     void mouseDoubleClickEvent(QMouseEvent* event) override;
00042     void mouseReleaseEvent(QMouseEvent* event) override;
00043     void resetSelection();
00044     void updateShapeDisplayCoords(Shape* item, const QPoint& position) const;
00046
00052     const alpha::vector<Shape*>& getShapes() const;
00053     int  getShapeSelected() const;
00054     int  getShapeSelectedIndex() const;
00056
00062     void setRenderShapes(const alpha::vector<Shape*>* renderShapes);
00063     void setShapeSelectedIndex(int newIndex);
00064     void setEditPrivileges(bool edit);
00066
00067 protected:
00073     void paintEvent(QPaintEvent *event) override;
00074
00075 private:
00081     const alpha::vector<Shape*>* renderShapes;
00082     int shapeSelectedIndex;
00083     bool allowEditing = false;
00085 };
```

9.49 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/resources.qrc File Reference

9.50 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.cpp File Reference

```
#include "TestimonialDialog.h"
#include "../backend/TestimonialManager.h"
```

Include dependency graph for TestimonialDialog.cpp:

9.51 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.h File Reference

Defines the TestimonialDialog class for submitting user testimonials.

```
#include <QDialog>
#include <QCheckBox>
#include <QTextEdit>
#include <QLineEdit>
#include <QVBoxLayout>
#include <QHBoxLayout>
#include <QPushButton>
#include <QLabel>
#include <QScrollArea>
```

Include dependency graph for TestimonialDialog.h:

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

Classes

- class TestimonialDialog

  The TestimonialDialog class provides a user interface for submitting testimonials.

9.51.1 Detailed Description

Defines the TestimonialDialog class for submitting user testimonials.

This dialog allows users to enter their name and testimonial content, and choose whether to be prompted again in the future.

9.52 TestimonialDialog.h

Go to the documentation of this file.

```
00001
00007 #ifndef TESTIMONIALDIALOG_H
00008 #define TESTIMONIALDIALOG_H
00009
00010 #include <QDialog>
00011 #include <QCheckBox>
00012 #include <QTextEdit>
00013 #include <QLineEdit>
00014 #include <QVBoxLayout>
00015 #include <QHBoxLayout>
00016 #include <QPushButton>
00017 #include <QLabel>
00018 #include <QScrollArea>
00019
00026 class TestimonialDialog : public QDialog {
00027     Q_OBJECT
00028
00029 public:
00034     explicit TestimonialDialog(QWidget* parent = nullptr);
00035
00036 private slots:
00043     void onSubmit();
00044     void onCancel();
00046
00047 private:
00055     QLineEdit* m_authorEdit;
00056     QTextEdit* m_contentEdit;
00057     QCheckBox* m_doNotShowAgain;
00059 };
00060
00061 #endif // TESTIMONIALDIALOG_H
```

9.53 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-↩ Modeler/src/frontend/TestimonialsDisplayDialog.cpp File Reference

```
#include "TestimonialsDisplayDialog.h"
#include "../backend/TestimonialManager.h"
```

Include dependency graph for TestimonialsDisplayDialog.cpp:

9.54 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-↩ Modeler/src/frontend/TestimonialsDisplayDialog.h File Reference

Defines the TestimonialsDisplayDialog class for showing customer testimonials.

```
#include <QDialog>
#include <QVBoxLayout>
#include <QScrollArea>
#include <QPushButton>
#include <QLabel>
#include <QFrame>
#include "TestimonialDialog.h"
```

Include dependency graph for TestimonialsDisplayDialog.h:

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

Classes

- class TestimonialsDisplayDialog

  The TestimonialsDisplayDialog class provides a UI to display customer testimonials.

9.54.1 Detailed Description

Defines the TestimonialsDisplayDialog class for showing customer testimonials.

This dialog displays a list of testimonials and provides an option for users to submit their own experiences.

9.55 TestimonialsDisplayDialog.h

Go to the documentation of this file.

```
00001
00007 #ifndef TESTIMONIALSDISPLAYDIALOG_H
00008 #define TESTIMONIALSDISPLAYDIALOG_H
00009
00010 #include <QDialog>
00011 #include <QVBoxLayout>
00012 #include <QScrollArea> // Included for context, used in .cpp
00013 #include <QPushButton> // Included for context, used in .cpp
00014 #include <QLabel>      // Included for context, used in .cpp
00015 #include <QFrame>      // Included for context, used in .cpp
00016 #include "TestimonialDialog.h"
00017
00024 class TestimonialsDisplayDialog : public QDialog {
00025     Q_OBJECT
00026
00027 public:
00032     explicit TestimonialsDisplayDialog(QWidget* parent = nullptr);
00033
00034 private slots:
00035     // No explicit slots are defined in the header, but a lambda is used in the .cpp
00036     // for the "Share Your Experience" button.
00037
00038 private:
00044     void refreshTestimonials();
00045
00050     QVBoxLayout* m_testimonialsLayout;
00051 };
00052
00053 #endif // TESTIMONIALSDISPLAYDIALOG_H
```

9.56 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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:

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

Enumerations

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

9.56.1 Enumeration Type Documentation

9.56.1.1 ShapeIDs

```
enum ShapeIDs
```

Enumerator

LINE	
POLYLINE	
POLYGON	
RECTANGLE	
SQUARE	
ELLIPSE	
CIRCLE	
TEXT	

9.57 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
```

9.58 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.cpp File Reference

`#include "circle.h"`

Include dependency graph for circle.cpp:

9.59 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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.

9.60 circle.h

Go to the documentation of this file.

```
#ifndef CIRCLE_H
#define CIRCLE_H

#include "shape.h"


class Circle : public Shape
{
public:
    Circle(string shapeType,
           QPoint coords,
           QPen   pen,
           QBrush brush,
           int r);

    void Draw(QWidget* renderArea) override;

    double Perimeter() const override;

    double Area()      const override;

    bool isPointInside(const QPoint& point) const override;

    int getR() const;

    void setR(int radius);
    void setX(int x);
    void setY(int y);

private:
    int r;
};

#endif // CIRCLE_H

```

9.61 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.cpp File Reference

`#include "ellipse.h"`

Include dependency graph for ellipse.cpp:

9.62 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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.

9.63 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;
```

```
    double Perimeter() const override;

    double Area()      const override;

    bool isPointInside(const QPoint& point) const override;

    int getA() const;
    int getB() const;

    void setA(int newA);
    void setB(int newB);
    void setX(int newX);
    void setY(int newY);

private:
    int a;
    int b;
};

#endif // ELLIPSE_H
```

9.64 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.cpp File Reference

`#include "line.h"`

Include dependency graph for line.cpp:

9.65 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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.

9.66 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;
00066
00068     void setStartPoint(const QPoint& newStartPoint);
00069     void setEndPoint(const QPoint& newEndPoint);
00070     void setX(int newX);
00071     void setY(int newY);
00072
00073 private:
00074     QPoint startPoint;
00075     QPoint endPoint;
00076 };
00077
00078 #endif // LINE_H
```

9.67 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.cpp File Reference

`#include "polygon.h"`

Include dependency graph for polygon.cpp:

9.68 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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.

9.69 polygon.h

Go to the documentation of this file.

```
#ifndef POLYGON_H
#define POLYGON_H

#include "shape.h"

class Polygon : public Shape
{
public:
    Polygon(string shapeType,
            QPoint coords,
            QPen   pen,
            QBrush brush,
            QPolygon pointsList);

    void Draw(QWidget* renderArea) override;

    void Move(int x, int y) override;

    double Perimeter() const override;

    double Area() const override;

    bool isPointInside(const QPoint& point) const override;

    QPolygon getPointsList() const;


    void setPointsList(const QPolygon& newPointsList);
    void setX(int newX);
    void setY(int newY);

private:
    QPolygon pointsList;
};
#endif // POLYGON_H
```

9.70 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-↩ Modeler/src/objects/polyline.cpp File Reference

`#include "polyline.h"`

Include dependency graph for polyline.cpp:

9.71 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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

  The Polyline class.

9.72 polyline.h

Go to the documentation of this file.

```
00001 #ifndef POLYLINE_H
00002 #define POLYLINE_H
00003
00004 #include "shape.h"
00005
00012 class Polyline : public Shape
00013 {
00014 public:
00023     Polyline(string shapeType,
00024              QPoint coords,
00025              QPen   pen,
00026              QBrush brush,
00027              QPolygon pointsList);
00028
00033     void Draw(QWidget* renderArea) override;
00034
```

```
    void Move(int x, int y) override;

    double Perimeter() const override;

    double Area() const override {return 0;}

    bool isPointInside(const QPoint& point) const override;

    QPolygon getPointsList() const;


    void setPointsList(const QPolygon& newPointsList);
    void setX(int newX);
    void setY(int newY);

private:
    QPolygon pointsList;
};

#endif // POLYLINE_H
```

9.73 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.cpp File Reference

`#include "rectangle.h"`

Include dependency graph for rectangle.cpp:

9.74 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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

  The Rectangle class.

9.75 rectangle.h

Go to the documentation of this file.

```
00001 #ifndef RECTANGLE_H
00002 #define RECTANGLE_H
00003
00004 #include "shape.h"
00005
00012 class Rectangle : public Shape
00013 {
00014 public:
00024     Rectangle(string shapeType,
00025               QPoint coords,
00026               QPen   pen,
00027               QBrush brush,
00028               int length,
00029               int width);
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 getLength() const;
00058     int getWidth()  const;
00059
00061     void setLength(int newLength);
00062     void setWidth(int newWidth);
00063     void setX(int newX);
00064     void setY(int newY);
00065
00066 private:
00067     int length;
00068     int width;
00069 };
00070
00071 #endif // RECTANGLE_H
00072
```

9.76 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.cpp File Reference

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

Include dependency graph for shape.cpp:

Functions

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

9.76.1 Function Documentation

9.76.1.1 operator<()

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

Parameters

shape1	
shape2	

Returns

9.76.1.2 operator==()

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

Parameters

shape1	
shape2	

Returns

9.77 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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 <QComboBox>
#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

9.77.1 Variable Documentation

9.77.1.1 PI

```
const double PI = 3.14
```

9.78 shape.h

Go to the documentation of this file.

```
#ifndef SHAPE_H
#define SHAPE_H

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


using std::string;

using Qt::GlobalColor;
using Qt::PenCapStyle;
using Qt::PenStyle;
using Qt::PenJoinStyle;
using Qt::BrushStyle;
using Qt::AlignmentFlag;

const double PI = 3.14;


class Shape
{
    friend bool operator==(const Shape& shape1, const Shape& shape2);
    friend bool operator<(const Shape& shape1, const Shape& shape2);

public:

    Shape(int shapeId,
          string shapeType,
          QPoint coords,
          QPen   pen,
          QBrush brush);

    virtual ~Shape();

    static int nextTracker[9];
    static bool trackersInUse[9000];

    virtual void Draw(QWidget* renderArea) = 0;

    virtual void Move(int x, int y);

    virtual double Perimeter() const = 0;

    virtual double Area()      const = 0;

    virtual bool isPointInside(const QPoint& point) const = 0;

    void CreateParentItem();

    void CreatePenChild();

    void CreateBrushChild();

    void CreatePointsChild(const int POINTS_NUM);

    int    getShapeId()   const;
    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 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, PenJoinStyle penJoinStyle);
    void setBrush(GlobalColor brushColor, BrushStyle brushStyle);

    // These functions make it easier to change pen and brush properties in RenderAreaManager::modifyShape()
    QPen& setInternalPen();
    QBrush& setInternalBrush();

protected:
    QTreeWidgetItem* parentItem;
    alpha::vector<QTreeWidgetItem*> childItems;
    alpha::vector<QTreeWidgetItem*> pointsItems;
    alpha::vector<QTreeWidgetItem*> penItems;
    alpha::vector<QTreeWidgetItem*> brushItems;

private:
    const int shapeId;
    int      trackerId;
    string   shapeType;

    QPen     pen;
    QBrush   brush;
    QPoint   coords;

    QPainter painter;

    bool isSelected = false;

    // Disable Copy Operations
    Shape(Shape& shape) = delete;
    Shape& operator=(Shape& object) = delete;
};

#endif // SHAPE_H
```

9.79 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.cpp File Reference

`#include "square.h"`

Include dependency graph for square.cpp:

9.80 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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

  The Square class.

9.81 square.h

Go to the documentation of this file.

```
#ifndef SQUARE_H
#define SQUARE_H

#include "shape.h"

class Square : public Shape
{
public:
    Square(string shapeType,
           QPoint coords,
           QPen   pen,
           QBrush brush,
           int length);

    void Draw(QWidget* renderArea) override;

    double Perimeter() const override;

    double Area()      const override;

    bool isPointInside(const QPoint& point) const override;

    int getLength() const;

    void setLength(int newLength);
    void setX(int newX);
    void setY(int newY);

private:
    int length;
};

#endif // SQUARE_H
```

9.82 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.cpp File Reference

`#include "text.h"`

Include dependency graph for text.cpp:

9.83 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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

  The Text class.

9.84 text.h

Go to the documentation of this file.

```
00001 #ifndef TEXT_H
00002 #define TEXT_H
00003
00004 #include "shape.h"
00005
00012 class Text : public Shape
00013 {
00014 public:
00026     Text(string shapeType,
00027          QPoint coords,
00028          QString textString,
00029          GlobalColor   textColor,
00030          AlignmentFlag textAlignment,
00031          QFont font,
00032          int length,
00033          int width);
```

```
00034
00039     void Draw(QWidget* renderArea) override;
00040
00045     double Perimeter() const override;
00046
00051     double Area()      const override;
00052
00058     bool isPointInside(const QPoint& point) const override;
00059
00061     int            getLength()        const;
00062     int            getWidth()         const;
00063     QString        getTextString()    const;
00064     GlobalColor    getTextColor()     const;
00065     QFont          getFont()          const;
00066     AlignmentFlag getTextAlignment() const;
00067     int            getFontStyle()     const;
00068     QFont::Weight getFontWeight()    const;
00069
00071     void setText(QString text);
00072     void setLength(int newLength);
00073     void setWidth(int newWidth);
00074     void setX(int newX);
00075     void setY(int newY);
00076     void setAlignment(Qt::AlignmentFlag alignment);
00077     QFont& setInternalFont();
00078
00079
00080 private:
00081     int length;
00082     int width;
00083
00084     QString        textString;
00085     GlobalColor    textColor;
00086     QFont          font;
00087     AlignmentFlag textAlignment;
00088 };
00089
00090 #endif // TEXT_H
```

9.85 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/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

9.85.1 Macro Definition Documentation

9.85.1.1 ALPHA_VECTOR_H

```
#define ALPHA_VECTOR_H
```

9.86 vector.h

Go to the documentation of this file.

```
#ifndef ALPHA_VECTOR_H
#define ALPHA_VECTOR_H  //This is necessary for inclusion guards. DO NOT DELETE
/*******************************************************************************
 * vector.h
 * -----------------------------------------------------------------------------
 * Worked on by: Aram, Aspen, Luke
 * -----------------------------------------------------------------------------
 * This vector will be used to hold the shapes that will be displayed.
 * It supports the following basic operations:
 *      - constructors for one or more arguments
 *      - default constructors
 *      - copy constructor
 *      - copy assignment
 *      - move constructor
 *      - move assignment
 *      - destructor
 *
 * Vector also supports:
 *      - a basic iterator member type and member function
 *      - begin()
 *      - end() operations
*******************************************************************************/
namespace alpha {
template<class T>

class vector
{
    int size_v;     // the size
    T* elem;        // a pointer to the elements
    int space;      // size + free_space

    public:
        /*********************************************************************
         * DEFAULT CONSTRUCTOR
         *********************************************************************/
        vector() : size_v(0), space(1) {
            elem = new T[space];
        } // END vector()

        /*********************************************************************
         * ALTERNATE CONSTRUCTOR - size given
         *********************************************************************/
        explicit vector(int s) : size_v(s), space(s) {
            elem = new T[space];
        } // END explicit vector(int s)

        /*********************************************************************
         * COPY CONSTRUCTOR
         *********************************************************************/
        vector(const vector& other) : size_v(other.size_v), space(other.space) {
            elem = new T[space];

            for (int i = 0; i < size_v; i++) {
                elem[i] = other.elem[i];
            } // END for (int i = 0; i < size_v; i++)
        } // END vector(const vector& other)

        /*********************************************************************
         * COPY ASSIGNMENT
         *********************************************************************/
```

```
        vector& operator=(const vector& other) {
            /*******************************************************************
            * CHECKS IF SELF-ASSIGNMENT
            *******************************************************************/
            if (this == &other) {
                return *this;
            }

            /*******************************************************************
            * IF NOT SELF-ASSIGNMENT
            *******************************************************************/
            delete[] elem;

            size_v = other.size_v;
            space = other.space;
            elem = new T[space];

            for (int i = 0; i < size_v; i++) {
                elem[i] = other.elem[i];
            }

            return *this;
        } // END vector& operator=vector

        /***********************************************************************
         * MOVE CONSTRUCTOR
         ***********************************************************************/
        vector(vector&& other) noexcept
            : size_v(other.size_v), elem(other.elem), space(other.space) {
            other.size_v = 0;
            other.elem = nullptr;
            other.space = 0;
        }

        /***********************************************************************
         * MOVE ASSIGNMENT
         ***********************************************************************/
        vector& operator=(vector&& other) noexcept {
            /*******************************************************************
            * CHECKS IF SELF-ASSIGNMENT
            *******************************************************************/
            if (this == &other) {
                return *this;
            }

            /*******************************************************************
            * IF NOT SELF-ASSIGNMENT
            *******************************************************************/
            delete[] elem;

            size_v = other.size_v;
            space = other.space;
            elem = other.elem;

            other.size_v = 0;
            other.space = 0;
            other.elem = nullptr;

           return *this;
        } // END vector& operator=(const vector&& other) noexcept

        /***********************************************************************
         * DESTRUCTOR
         ***********************************************************************/
        ~vector() { delete[] elem; }

        /***********************************************************************
         * ACCESSOR - RETURN REFERENCE - MODIFIABLE
         ***********************************************************************/
        T& operator[] (int n) { return elem[n]; }

        /***********************************************************************
         * ACCESSOR - RETURN REFERENCE
         ***********************************************************************/
        const T& operator[] (int n) const { return elem[n]; }

        /***********************************************************************
         * ACCESSOR - RETURN CURRENT SIZE
         ***********************************************************************/
        int size() const { return size_v; }

        /***********************************************************************
         * ACCESSOR - RETURN CURRENT AVAILABLE SPACE
         ***********************************************************************/
        int capacity() const { return space; }

        /***********************************************************************
```

```
 * void resize(int newsize)
 * ------------------------------------------------------------------------
 * This function will be passed a number:
 *      newsize     - size to increase vector by
 *
 * depending on size_v the following will happen:
 *      size_v = newsize
 *      nothing will happen
 *
 *      size_v < newsize
 *      will change size_v to newsize
 *
 *      size_v > newsize (default)
 *      error will occur
 * -----------------------------------------------------------------------
 * PRE-CONDITIONS
 *      size_v  - original size of vector
 *
 * POST-CONDITIONS
 *      newsize  - new size of vector
 ************************************************************************/
void resize(int newsize) {

    /***********************************************************************
    * DOES NOTHING - EQUALS EACH OTHER
    ***********************************************************************/
    if (size_v == newsize) {}

    /***********************************************************************
    * OG SIZE LESS THAN NEW SIZE
    ***********************************************************************/
    else if (size_v < newsize) {
        reserve(newsize);
        size_v = newsize;
    } // END else if (size_v < newsize)

    /***********************************************************************
    * DOES NOTHING - DEFAULT - SMALLER THAN OG / INVALID INPUT / ERROR
    ***********************************************************************/
    else { }

} // END void resize(int newsize)

/************************************************************************
 * FUNCTION - ADD ELEMENT
 * ------------------------------------------------------------------------
 * This function will be passed a value:
 *      val     - data to add to vector
 *
 * Function adds the data to the back of the vector. If there is no
 * space then the function will increase the space before adding
 * -----------------------------------------------------------------------
 * PRE-CONDITIONS
 *      size_v  - original size of vector
 *
 * POST-CONDITIONS
 *      newsize  - new size of vector
 ************************************************************************/
void push_back(const T val) {
    /***********************************************************************
     * IF NO SPACE - MAKE ROOM
     ***********************************************************************/
    if (size_v == space) {
        reserve(space + 1);
    }
    /***********************************************************************
     * IF SPACE
     ***********************************************************************/
    elem[size_v] = val;
    size_v++;
} // END void push_back(const T val)

 /***********************************************************************
 * void reserve(int newalloc)
 * ------------------------------------------------------------------------
 * This function will be passed a number:
 *      newalloc     - size to increase vector capacity by
 *
 * depending on space the following will happen:
 *      space = newalloc
 *      nothing will happen
 *
 *      space < newalloc
 *      will change space to newalloc
 *
 *      size_v > newalloc (default)
 *      error will occur
```

```
 * ---------------------------------------------------------------------
 * PRE-CONDITIONS
 *      space  - original size of vector
 *
 * POST-CONDITIONS
 *      newalloc  - new size of vector
 *********************************************************************/
void reserve(int newalloc) {
    /*********************************************************************
    * DOES NOTHING - EQUALS EACH OTHER
    *********************************************************************/
    if (space == newalloc) {}

    /*********************************************************************
    * OG SIZE LESS THAN NEW SIZE
    *********************************************************************/
    else if (space < newalloc) {

        // CREATE BIGGER ARRAY
        T* new_elem = new T[newalloc];

        // COPY DATA
        for (int i = 0; i < size_v; i++) {
            new_elem[i] = elem[i];
        }

        delete[] elem;
        elem = new_elem;

        // UPDATE SPACE
        space = newalloc;
    } // END else if (space < newalloc)

    /*********************************************************************
    * DOES NOTHING - DEFAULT - SMALLER THAN OG / INVALID INPUT / ERROR
    *********************************************************************/
    else { }

} // END void reserve(int newalloc)


using iterator = T*;
using const_iterator = const T*;

/*********************************************************************
 * POINTS TO FIRST ELEMENT
 *********************************************************************/
iterator begin() { return elem; }

/*********************************************************************
 * CONSTANT - POINTS TO FIRST ELEMENT
 *********************************************************************/
const_iterator begin() const { return elem; }

/*********************************************************************
 * POINTS TO ONE BEYOND THE LAST ELEMENT
 *********************************************************************/
iterator end() { return elem + size_v; }

/*********************************************************************
 * CONSTANT - POINTS TO ONE BEYOND THE LAST ELEMENT
 *********************************************************************/
const_iterator end() const { return elem + size_v; }

/*********************************************************************
 * INSERT NEW ELEMENT (V) BEFORE P
 *********************************************************************/
iterator insert(iterator p, const T& v) {
    /*********************************************************************
     * CHECK IF ENOUGH SPACE
     *********************************************************************/
    // record the index where we want to insert
    int index = static_cast<int>(p - elem);

    // grow storage if needed
    if (size_v == space) {
        reserve(space + 1);
    }

    // iterator next = end();
    // while (next != p) {
    //     *next = *(next - 1);
    //     next--;
    // }

    // recompute pointers into the (possibly moved) data
    p = elem + index;
```

```
            iterator end_it = elem + size_v;

            // shift elements [index..size_v-1] one position to the right
            for (iterator it = end_it; it != p; --it)
                *it = *(it - 1);

            *p = v;
            size_v++;

            return p;
        } // END iterator insert(iterator p, const T& v)

        /*************************************************************************
         * REMOVE ELEMENT POINTED TO BY P
         *************************************************************************/
        iterator erase(iterator p) {
            /*********************************************************************
             * IF P AT THE END
             *********************************************************************/
            if (p == end()) {
                return p;
            }

            iterator next = p + 1;
            while (next != end()) {
                *p = *next;
                p++;
                next++;
            }

            p--;
            p->~T();
            size_v--;

            return p;
        } // END iterator erase(iterator p)

}; // END class vector
}; // END namespace alpha
#endif // ALPHA_VECTOR_H
```

9.87 /Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/webservice/webservice.cpp File Reference

Implements the Crow web service for handling shapes, render area, user, and testimonial 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 ()

9.87.1 Detailed Description

Implements the Crow web service for handling shapes, render area, user, and testimonial data.

This web service provides the following endpoints, grouped by entity:

- TEST GET "/" : A test endpoint that returns a greeting.
- Shapes Endpoints:
- GET /shapes : Returns the JSON-formatted shapes data from ../../database/shapes.json.
- POST /shapes : Accepts JSON data to update the shapes file (../../database/shapes.json).
- DELETE /shapes-all : Clears all shapes data by truncating the shapes.json file.
- Render Area Endpoints:
- GET /render_area : Returns the JSON-formatted render area data from ../../database/render_area.json.
- POST /render_area : Accepts JSON data to update the render area file (../../database/render_area.json).
- DELETE /render_area-all : Clears all render area data by truncating the render_area.json file.
- Users Endpoints:
- GET /users : Returns the JSON-formatted user data from ../../database/users.json.
- POST /users : Accepts JSON data to update the user file (../../database/users.json).
- DELETE /users-all : Clears all user data by truncating the users.json file.
- Testimonials Endpoints:
- GET /testimonials : Returns JSON-formatted testimonial data from ../../database/testimonials.json.
- POST /testimonials : Accepts JSON data to update the testimonials file (../../database/testimonials.json).
- DELETE /testimonials : Clears all testimonial data by truncating the testimonials.json file. (Note: Original was DELETE /testimonials, implying delete all)

Note

Uses the Crow framework for handling HTTP requests. Make sure the database directory exists or is created.

This web service provides the following endpoints, grouped by entity:

- TEST GET "/" : A test endpoint that returns a greeting.
- Shapes Endpoints:
- GET /shapes : Returns the JSON-formatted shapes data from /webservice/database/shapes.json.
- POST /shapes : Accepts JSON data to update the shapes file (/webservice/database/shapes.json).

- DELETE /shapes-all : Clears all shapes data by truncating the shapes.json file.
- Render Area Endpoints:
- GET /render_area : Returns the JSON-formatted render area data from /webservice/database/render_↩area.json.
- POST /render_area : Accepts JSON data to update the render area file (/webservice/database/render_↩area.json).
- DELETE /render_area-all : Clears all render area data by truncating the render_area.json file.
- Users Endpoints:
- GET /users : Returns the JSON-formatted user data from /webservice/database/users.json.
- POST /users : Accepts JSON data to update the user file (/webservice/database/users.json).
- DELETE /users-all : Clears all user data by truncating the users.json file.
- Testimonials Endpoints:
- GET /testimonials : Returns JSON-formatted testimonial data from /webservice/database/testimonials.json.
- POST /testimonials : Accepts JSON data to update the testimonials file (/webservice/database/testimonials.json).
- DELETE /testimonials : Clears all testimonial data by truncating the testimonials.json file.

Note

Uses the Crow framework for handling HTTP requests. Make sure the database directory exists or is created.

9.87.2 Function Documentation

9.87.2.1 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:

9.87.2.2 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".

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.

Clears all shapes.

Deletes the shapes.json file content by truncating the file, leaving it empty (an empty JSON array).

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".

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.

Clears all render area data.

Deletes the render_area.json file content by truncating the file, leaving it empty (an empty JSON array).

Retrieves user account data.

Reads the JSON data from "../../database/users.json", parses it, and returns it with Content-Type "application/json".

Updates users.json with new user data.

Accepts a POST request with JSON body and writes it to "../../database/users.json".

Clears all user account data.

Deletes the users.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
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.cpp, 351
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ApiClient.h, 351, 352
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.cpp, 353
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/AppDriver.h, 353, 354
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.cpp, 357
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Parser.h, 357, 358
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.cpp, 360
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/RenderAreaManager.h, 360, 361
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.cpp, 362
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/ShapesManager.h, 362, 363
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.cpp, 364
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/Testimonial.h, 364, 365
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.cpp, 366
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/TestimonialManager.h, 366, 367
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.cpp, 368
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserAccount.h, 368, 369
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.cpp, 370
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/UserManager.h, 370, 371
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/main.cpp, 376
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_ApiClient.cpp, 355
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/backend/moc_UserManager.cpp, 356
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/ColumnEditDelegate.h, 372, 373
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Geoo.qss, 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/Medize.qss, 388
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.cpp, 390
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialDialog.h, 391, 392
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialsDisplayDialog.cpp, 392
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/TestimonialsDisplayDialog.h, 393, 394
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/darkstyle.qss, 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/lightstyle.qss, 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.cpp, 374
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/loginwindow.h, 374, 375
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/main.cpp, 380
/Users/gautam/Local/Forge/Team-Alphawolves/2D-

Graphics-Modeler/src/frontend/mainwindow.cpp, 385
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.h, 385, 386
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/mainwindow.ui, 388
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.cpp, 388
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/renderarea.h, 389, 390
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/frontend/resources.qrc, 390
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/all_shapes.h, 394, 395
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.cpp, 396
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/circle.h, 396, 397
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.cpp, 397
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/ellipse.h, 398
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.cpp, 399
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/line.h, 399, 400
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.cpp, 401
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polygon.h, 401, 402
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.cpp, 402
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/polyline.h, 403
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.cpp, 404
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/rectangle.h, 404, 405
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.cpp, 406
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/shape.h, 407, 408
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.cpp, 410
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/square.h, 410, 411
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.cpp, 411
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/text.h, 412
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/objects/vector.h, 413, 414
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/src/webservice/webservice.cpp, 418
/Users/gautam/Local/Forge/Team-Alphawolves/2D-Graphics-Modeler/webservice-dockerized/main.cpp, 382
~AppDriver
AppDriver, 34
~MainWindow
MainWindow, 93
~Parser
Parser, 139
~RenderAreaManager
RenderAreaManager, 212
~Shape
Shape, 228
~ShapesManager
ShapesManager, 260
~TestimonialManager
TestimonialManager, 293
~UserAccount
UserAccount, 323
~UserManager
UserManager, 332
~vector
alpha::vector< T >, 345

a
Ellipse, 64
addShape
RenderAreaManager, 212
ShapesManager, 260
addTestimonial
TestimonialManager, 294
addToShapeTree
MainWindow, 94
addUser
UserManager, 332
admin
Parser::RawUser, 188
UserAccount, 327
all_shapes.h
CIRCLE, 395
ELLIPSE, 395
LINE, 395
POLYGON, 395
POLYLINE, 395
RECTANGLE, 395
ShapeIDs, 395

SQUARE, 395
TEXT, 395
allocateTrackerId
Shape, 229
allowEditing
RenderArea, 206
alpha, 13
alpha::vector< T >, 341
~vector, 345
begin, 346
capacity, 346
const_iterator, 344
elem, 350
end, 346
erase, 346
insert, 347
iterator, 344
operator=, 347
operator[], 348
push_back, 348
reserve, 349
resize, 349
size, 349
size_v, 350
space, 350
vector, 344, 345
ALPHA_VECTOR_H
vector.h, 414
AnalyzeDeleteReply
ApiClient, 20
AnalyzeGetReply
ApiClient, 21
AnalyzePostReply
ApiClient, 22
ApiClient, 15
AnalyzeDeleteReply, 20
AnalyzeGetReply, 21
AnalyzePostReply, 22
ApiClient, 20
BadDeleteReply, 23
BadGetReply, 23
BadPostReply, 24
DeleteRenderAreaAll, 24
DeleteShapesAll, 24
DeleteTestimonialsAll, 25
DeleteUsersAll, 25
GetRenderArea, 25
GetShapes, 26
GetTestimonials, 26
GetUsers, 26
GoodDeleteReply, 27
GoodGetReply, 27
GoodPostReply, 28
manager, 30
PostRenderArea, 28
PostShapes, 28
PostTestimonials, 29
PostUsers, 29
AppDriver, 30
~AppDriver, 34
AppDriver, 34
connectFrontendToDriver, 35
connectManagersToFrontend, 35
loadAllData, 37
mainWindow, 41
onDeleteAllUsers, 37
onLoginAttempt, 37
onNewUser, 37
onRenderDeleteAllShapes, 38
onRenderShapeAdded, 38
onRenderShapeChanged, 38
onRenderShapeDeleted, 39
onUserDeleted, 39
onUserModified, 39
renderedShapes, 41
run, 40
shutdown, 41
user, 41
AppendBrushData
Parser, 140
AppendCommonShapeData
Parser, 141
AppendTextData
Parser, 142
Area
Circle, 46
Ellipse, 59
Line, 69
Polygon, 171
Polyline, 181
Rectangle, 194
Shape, 229
Square, 273
Text, 311
attemptLogin
LoginWindow, 79
attemptSignup
LoginWindow, 79
authenticate
UserManager, 333
authenticationFailed
UserManager, 333

b
Ellipse, 64
backToLoginBtn
LoginWindow, 83
BadDeleteReply
ApiClient, 23
BadGetReply
ApiClient, 23
BadPostReply
ApiClient, 24
begin
alpha::vector< T >, 346
brush
Parser::MorphicShape, 133

Shape, 254
brushItems
Shape, 254
BuildShape
Parser, 144

canEdit
ColumnEditDelegate, 54
capacity
alpha::vector< T >, 346
checkTimeAndPrompt
TestimonialManager, 294
childItems
Shape, 254
CIRCLE
all_shapes.h, 395
Circle, 42
Area, 46
Circle, 45
Draw, 46
getR, 47
isPointInside, 47
Perimeter, 48
r, 50
setR, 48
setX, 49
setY, 49
client
RenderAreaManager, 222
ShapesManager, 267
TestimonialManager, 301
UserManager, 340
ColumnEditDelegate, 51
canEdit, 54
ColumnEditDelegate, 53
createEditor, 54
setCanEdit, 54
connectFrontendToDriver
AppDriver, 35
connectManagersToFrontend
AppDriver, 35
const_iterator
alpha::vector< T >, 344
contactUsWidget
MainWindow, 130
contactWindow
MainWindow, 130
coords
Parser::MorphicShape, 133
Shape, 254
createAlignmentComboBox
MainWindow, 95
CreateBrushChild
Shape, 230
createBrushStyleComboBox
MainWindow, 96
createColorComboBox
MainWindow, 96
createEditor
ColumnEditDelegate, 54
createFontComboBox
MainWindow, 97
createFontStyleComboBox
MainWindow, 97
createFontWeightComboBox
MainWindow, 98
CreateParentItem
Shape, 230
createPenCapStyleComboBox
MainWindow, 98
CreatePenChild
Shape, 232
createPenJoinStyleComboBox
MainWindow, 99
createPenStyleComboBox
MainWindow, 99
createPenWidthSpinBox
MainWindow, 100
CreatePointsChild
Shape, 232
createShapeTableTab
MainWindow, 100
createShapeTypeComboBox
MainWindow, 101
currUser
MainWindow, 130
UserManager, 340

delegate
MainWindow, 130
deleteAllShapes
MainWindow, 101
RenderAreaManager, 213
ShapesManager, 261
deleteAllUsers
UserManager, 334
DeleteRenderAreaAll
ApiClient, 24
deleteShape
RenderAreaManager, 213
ShapesManager, 261
DeleteShapesAll
ApiClient, 24
DeleteTestimonialsAll
ApiClient, 25
deleteUser
UserManager, 334
DeleteUsersAll
ApiClient, 25
displayedTextChanged
MainWindow, 101
Draw
Circle, 46
Ellipse, 59
Line, 69
Polygon, 171
Polyline, 181
Rectangle, 194

Shape, 233
Square, 273
Text, 311
drawShapes
MainWindow, 102

elem
alpha::vector< T >, 350
ELLIPSE
all_shapes.h, 395
Ellipse, 55
a, 64
Area, 59
b, 64
Draw, 59
Ellipse, 59
getA, 60
getB, 61
isPointInside, 61
Perimeter, 61
setA, 62
setB, 62
setX, 62
setY, 63
end
alpha::vector< T >, 346
endPoint
Line, 74
erase
alpha::vector< T >, 346
ExtractArray
Parser, 144
ExtractInteger
Parser, 145
ExtractKey
Parser, 146
ExtractLiteral
Parser, 146
ExtractValue
Parser, 147

font
Parser::MorphicShape, 133
Text, 319
fromJson
Testimonial, 280

get_json_file
main.cpp, 383
webservice.cpp, 420
getA
Ellipse, 60
GetAlignmentFlag
Parser, 148
getAuthor
Testimonial, 281
getB
Ellipse, 61
getBrush
Shape, 233
getBrushColor
Shape, 234
getBrushItems
Shape, 234
getBrushItemsEnd
Shape, 235
GetBrushStyle
Parser, 149
getBrushStyle
Shape, 235
getChildEnd
Shape, 236
getChildItems
Shape, 236
GetColor
Parser, 150
GetConnectedClient
main.cpp, 377
getContent
Testimonial, 281
getCurrUserRef
UserManager, 334
getDoNotShowAgain
TestimonialManager, 295
getEndPoint
Line, 70
getFont
Text, 312
GetFontStyle
Parser, 151
getFontStyle
Text, 312
GetFontWeight
Parser, 151
getFontWeight
Text, 313
getInstance
TestimonialManager, 295
getLength
Rectangle, 195
Square, 274
Text, 313
getPainter
Shape, 237
getParentItem
Shape, 238
getPassword
UserAccount, 324
getPen
Shape, 238
GetPenCapStyle
Parser, 152
getPenCapStyle
Shape, 239
getPenColor
Shape, 239
getPenItems

Shape, 240
getPenItemsEnd
Shape, 240
GetPenJoinStyle
Parser, 153
getPenJoinStyle
Shape, 241
GetPenStyle
Parser, 154
getPenStyle
Shape, 241
getPenWidth
Shape, 242
getPoints
Shape, 242
getPointsItems
Shape, 243
getPointsList
Polygon, 172
Polyline, 182
getR
Circle, 47
GetRenderArea
ApiClient, 25
GetRenderAreaTestString
main.cpp, 377
getSatisfactoryTestimonials
TestimonialManager, 296
getSelected
Shape, 243
GetShapeDimensions
Parser, 155
getShapeId
Shape, 244
GetShapes
ApiClient, 26
getShapes
RenderArea, 203
getShapeSelected
RenderArea, 203
getShapeSelectedIndex
RenderArea, 203
getShapesRef
RenderAreaManager, 214
ShapesManager, 261
GetShapeTestString
main.cpp, 378
getShapeType
Shape, 245
getStartPoint
Line, 70
GetTestimonials
ApiClient, 26
getTextAlignment
Text, 314
getTextColor
Text, 314
getTextString
Text, 314
getTimestamp
Testimonial, 281
getTrackerId
Shape, 245
getUsername
UserAccount, 324
GetUsers
ApiClient, 26
GetUsersTestString
main.cpp, 378
getWidth
Rectangle, 195
Text, 315
getX
Shape, 246
getY
Shape, 247
GoodDeleteReply
ApiClient, 27
GoodGetReply
ApiClient, 27
GoodPostReply
ApiClient, 28

hasAdmin
Parser::RawUser, 188
hasPassword
Parser::RawUser, 188
hasUserGivenTestimonial
TestimonialManager, 297
hasUsername
Parser::RawUser, 188

INITIAL_PROMPT_TIME
TestimonialManager, 301
insert
alpha::vector< T >, 347
isAdmin
UserAccount, 325
isGuest
Testimonial, 281
isPointInside
Circle, 47
Ellipse, 61
Line, 71
Polygon, 173
Polyline, 182
Rectangle, 196
Shape, 248
Square, 274
Text, 315
isSatisfactory
Testimonial, 281
isSelected
Shape, 254
iterator
alpha::vector< T >, 344

JsonToShapes
Parser, 157
JsonToTestimonials
Parser, 157
JsonToUsers
Parser, 158

length
Rectangle, 198
Square, 277
Text, 319
LINE
all_shapes.h, 395
Line, 64
Area, 69
Draw, 69
endPoint, 74
getEndPoint, 70
getStartPoint, 70
isPointInside, 71
Line, 69
Move, 71
Perimeter, 72
setEndPoint, 72
setStartPoint, 73
setX, 73
setY, 74
startPoint, 74
loadAllData
AppDriver, 37
loadShapes
RenderAreaManager, 214
ShapesManager, 262
loadStyleSheet
MainWindow, 102
loadTestimonials
TestimonialManager, 297
loadUsers
UserManager, 334
loginAttempt
MainWindow, 102
loginBtn
LoginWindow, 83
loginFailed
MainWindow, 102
loginPage
LoginWindow, 83
loginPassEdit
LoginWindow, 83
loginRequested
LoginWindow, 80
loginSuccess
MainWindow, 103
loginUserEdit
LoginWindow, 83
LoginWindow, 75
attemptLogin, 79
attemptSignup, 79
backToLoginBtn, 83
loginBtn, 83
loginPage, 83
loginPassEdit, 83
loginRequested, 80
loginUserEdit, 83
LoginWindow, 79
password, 81
showLoginPage, 81
showSignupPage, 81
signupBtn, 83
signupPage, 84
signupPassEdit, 84
signupRequested, 82
signupUserEdit, 84
stack, 84
toSignupBtn, 84
username, 82
logoLabel
MainWindow, 130

m_author
Testimonial, 282
m_authorEdit
TestimonialDialog, 287
m_content
Testimonial, 282
m_contentEdit
TestimonialDialog, 287
m_doNotShowAgain
TestimonialDialog, 287
TestimonialManager, 301
m_isGuest
Testimonial, 282
m_isSatisfactory
Testimonial, 282
m_testimonials
TestimonialManager, 301
m_testimonialsLayout
TestimonialsDisplayDialog, 305
m_timestamp
Testimonial, 282
m_trackingTimer
TestimonialManager, 301
m_userTimeTracking
TestimonialManager, 301
main
main.cpp, 378, 381, 383
webservice.cpp, 420
main.cpp
get_json_file, 383
GetConnectedClient, 377
GetRenderAreaTestString, 377
GetShapeTestString, 378
GetUsersTestString, 378
main, 378, 381, 383
OnBadGetResponseTest, 378
OnBadPostResponseTest, 379
OnGoodGetResponseTest, 379
OnGoodPostResponseTest, 379

pApp, 380
parse, 380
pClient, 380
MainWindow, 85
~MainWindow, 93
addToShapeTree, 94
contactUsWidget, 130
contactWindow, 130
createAlignmentComboBox, 95
createBrushStyleComboBox, 96
createColorComboBox, 96
createFontComboBox, 97
createFontStyleComboBox, 97
createFontWeightComboBox, 98
createPenCapStyleComboBox, 98
createPenJoinStyleComboBox, 99
createPenStyleComboBox, 99
createPenWidthSpinBox, 100
createShapeTableTab, 100
createShapeTypeComboBox, 101
currUser, 130
delegate, 130
deleteAllShapes, 101
displayedTextChanged, 101
drawShapes, 102
loadStyleSheet, 102
loginAttempt, 102
loginFailed, 102
loginSuccess, 103
logoLabel, 130
MainWindow, 92, 93
newUserAdded, 103
on_actionnew_circle_button_triggered, 103
on_actionnew_ellipse_button_triggered, 104
on_actionnew_line_button_triggered, 105
on_actionnew_polygon_button_triggered, 106
on_actionnew_polyline_button_triggered, 107
on_actionnew_rectange_button_triggered, 108
on_actionnew_square_button_triggered, 109
on_actionnew_text_button_triggered, 110
on_actionremove_shape_button_triggered, 111
onComboBoxChanged, 112
onContactUsClicked, 112
onLoginClicked, 113
onLoginRequest, 113
onRenderAreaChanged, 114
onRenderAreaNotChanged, 116
onSignupRequest, 116
onSortMethodChanged, 116
onSpinBoxChanged, 117
onToggleStyle, 118
onTreeWidgetItemChanged, 118
onUserAuthentication, 119
onUserAuthenticationFailure, 121
populateShapeTable, 121
renderArea, 130
renderShapes, 130
selection_sort, 121
setupTestimonials, 122
shapeAdded, 123
shapeChanged, 124
shapeDeleted, 124
shapes_to_treeWidget, 125
shapeTable, 130
showRenderStatusMessage, 127
showTestimonialPrompt, 127
showTestimonialsDisplay, 127
sortByArea, 128
sortById, 128
sortByPerimeter, 129
sortDropdown, 131
sortOrderDropdown, 131
tabWidget, 131
teamNameLabel, 131
ui, 131
userStatusLabel, 131
mainWindow
AppDriver, 41
manager
ApiClient, 30
moc_ApiClient.cpp
Q_CONSTINIT, 356
qt_meta_data_ZN9ApiClientE, 356
moc_UserManager.cpp
Q_CONSTINIT, 357
qt_meta_data_ZN11UserManagerE, 357
modifyDisplayedText
RenderAreaManager, 214
modifyShape
RenderAreaManager, 215
ShapesManager, 262
modifyUser
UserManager, 335
mouseDoubleClickEvent
RenderArea, 203
mouseMoveEvent
RenderArea, 203
mousePressEvent
RenderArea, 204
mouseReleaseEvent
RenderArea, 204
Move
Line, 71
Polygon, 173
Polyline, 183
Shape, 249

newUserAdded
MainWindow, 103
nextTracker
Shape, 254

on_actionnew_circle_button_triggered
MainWindow, 103
on_actionnew_ellipse_button_triggered
MainWindow, 104
on_actionnew_line_button_triggered

MainWindow, 105
on_actionnew_polygon_button_triggered
MainWindow, 106
on_actionnew_polyline_button_triggered
MainWindow, 107
on_actionnew_rectange_button_triggered
MainWindow, 108
on_actionnew_square_button_triggered
MainWindow, 109
on_actionnew_text_button_triggered
MainWindow, 110
on_actionremove_shape_button_triggered
MainWindow, 111
onBadDeleteResponse
RenderAreaManager, 216
ShapesManager, 262
UserManager, 335
onBadGetResponse
RenderAreaManager, 217
ShapesManager, 263
TestimonialManager, 297
UserManager, 336
OnBadGetResponseTest
main.cpp, 378
onBadPostResponse
RenderAreaManager, 218
ShapesManager, 263
TestimonialManager, 297
UserManager, 336
OnBadPostResponseTest
main.cpp, 379
onCancel
TestimonialDialog, 286
onComboBoxChanged
MainWindow, 112
onContactUsClicked
MainWindow, 112
onDeleteAllUsers
AppDriver, 37
onGoodDeleteResponse
RenderAreaManager, 218
ShapesManager, 264
UserManager, 337
onGoodGetResponse
RenderAreaManager, 219
ShapesManager, 264
TestimonialManager, 298
UserManager, 337
OnGoodGetResponseTest
main.cpp, 379
onGoodPostResponse
RenderAreaManager, 220
ShapesManager, 265
TestimonialManager, 298
UserManager, 338
OnGoodPostResponseTest
main.cpp, 379
onLoginAttempt
AppDriver, 37
onLoginClicked
MainWindow, 113
onLoginRequest
MainWindow, 113
onNewUser
AppDriver, 37
onRenderAreaChanged
MainWindow, 114
onRenderAreaNotChanged
MainWindow, 116
onRenderDeleteAllShapes
AppDriver, 38
onRenderShapeAdded
AppDriver, 38
onRenderShapeChanged
AppDriver, 38
onRenderShapeDeleted
AppDriver, 39
onSignupRequest
MainWindow, 116
onSortMethodChanged
MainWindow, 116
onSpinBoxChanged
MainWindow, 117
onSubmit
TestimonialDialog, 286
onToggleStyle
MainWindow, 118
onTreeWidgetItemChanged
MainWindow, 118
onUserAuthentication
MainWindow, 119
onUserAuthenticationFailure
MainWindow, 121
onUserDeleted
AppDriver, 39
onUserModified
AppDriver, 39
operator<
Shape, 253
shape.cpp, 406
operator=
alpha::vector< T >, 347
Parser, 159
Shape, 249
UserAccount, 325
operator==
Shape, 253
shape.cpp, 406
operator[]
alpha::vector< T >, 348

painter
Shape, 254
paintEvent
RenderArea, 204
pApp
main.cpp, 380

parentItem
Shape, 255
parse
main.cpp, 380
RenderAreaManager, 222
ShapesManager, 267
TestimonialManager, 301
UserManager, 341
ParseJsonObject
Parser, 160
Parser, 135
~Parser, 139
AppendBrushData, 140
AppendCommonShapeData, 141
AppendTextData, 142
BuildShape, 144
ExtractArray, 144
ExtractInteger, 145
ExtractKey, 146
ExtractLiteral, 146
ExtractValue, 147
GetAlignmentFlag, 148
GetBrushStyle, 149
GetColor, 150
GetFontStyle, 151
GetFontWeight, 151
GetPenCapStyle, 152
GetPenJoinStyle, 153
GetPenStyle, 154
GetShapeDimensions, 155
JsonToShapes, 157
JsonToTestimonials, 157
JsonToUsers, 158
operator=, 159
ParseJsonObject, 160
Parser, 139
PrintShapeVector, 161
ShapesToJson, 161
SkipWhitespace, 162
StringToVector, 163
TestimonialsToJson, 163
UpdateAccumulator, 164
UpdateUserAccumulator, 165
UsersToJson, 165
Parser::MorphicShape, 132
brush, 133
coords, 133
font, 133
pen, 133
shapeDimensions, 134
shapeId, 134
shapeType, 134
textAlignment, 134
textColor, 134
textString, 134
trackerId, 134
Parser::RawUser, 187
admin, 188
hasAdmin, 188
hasPassword, 188
hasUsername, 188
password, 188
username, 189
password
LoginWindow, 81
Parser::RawUser, 188
UserAccount, 327
pClient
main.cpp, 380
pen
Parser::MorphicShape, 133
Shape, 255
penItems
Shape, 255
Perimeter
Circle, 48
Ellipse, 61
Line, 72
Polygon, 174
Polyline, 184
Rectangle, 196
Shape, 250
Square, 275
Text, 316
PI
shape.h, 408
pointsItems
Shape, 255
pointsList
Polygon, 176
Polyline, 186
POLYGON
all_shapes.h, 395
Polygon, 166
Area, 171
Draw, 171
getPointsList, 172
isPointInside, 173
Move, 173
Perimeter, 174
pointsList, 176
Polygon, 170
setPointsList, 175
setX, 175
setY, 176
POLYLINE
all_shapes.h, 395
Polyline, 177
Area, 181
Draw, 181
getPointsList, 182
isPointInside, 182
Move, 183
Perimeter, 184
pointsList, 186
Polyline, 180

setPointsList, 184
setX, 184
setY, 185
populateShapeTable
MainWindow, 121
PostRenderArea
ApiClient, 28
PostShapes
ApiClient, 28
PostTestimonials
ApiClient, 29
PostUsers
ApiClient, 29
PrintShapeVector
Parser, 161
push_back
alpha::vector< T >, 348

Q_CONSTINIT
moc_ApiClient.cpp, 356
moc_UserManager.cpp, 357
qt_meta_data_ZN11UserManagerE
moc_UserManager.cpp, 357
qt_meta_data_ZN9ApiClientE
moc_ApiClient.cpp, 356
QT_WARNING_DISABLE_DEPRECATED, 13
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN11UserManagerE_t, 186
QT_WARNING_DISABLE_DEPRECATED::qt_meta_tag_ZN9ApiClientE_t, 187

r
Circle, 50
RECTANGLE
all_shapes.h, 395
Rectangle, 189
Area, 194
Draw, 194
getLength, 195
getWidth, 195
isPointInside, 196
length, 198
Perimeter, 196
Rectangle, 194
setLength, 196
setWidth, 197
setX, 197
setY, 198
width, 198
refreshTestimonials
TestimonialsDisplayDialog, 304
RenderArea, 199
allowEditing, 206
getShapes, 203
getShapeSelected, 203
getShapeSelectedIndex, 203
mouseDoubleClickEvent, 203
mouseMoveEvent, 203
mousePressEvent, 204
mouseReleaseEvent, 204
paintEvent, 204
RenderArea, 203
renderShapes, 206
resetSelection, 205
setEditPrivileges, 205
setRenderShapes, 205
setShapeSelectedIndex, 205
shapeSelectedIndex, 207
updateShapeDisplayCoords, 205
renderArea
MainWindow, 130
renderAreaChanged
RenderAreaManager, 220
RenderAreaManager, 207
~RenderAreaManager, 212
addShape, 212
client, 222
deleteAllShapes, 213
deleteShape, 213
getShapesRef, 214
loadShapes, 214
modifyDisplayedText, 214
modifyShape, 215
onBadDeleteResponse, 216
onBadGetResponse, 217
onBadPostResponse, 218
onGoodDeleteResponse, 218
onGoodGetResponse, 219
onGoodPostResponse, 220
parse, 222
renderAreaChanged, 220
RenderAreaManager, 211
renderAreaNotChanged, 221
renderedShapes, 223
saveShapes, 221
statusMessage, 222
renderAreaNotChanged
RenderAreaManager, 221
renderedShapes
AppDriver, 41
RenderAreaManager, 223
renderShapes
MainWindow, 130
RenderArea, 206
REPEAT_PROMPT_TIME
TestimonialManager, 301
reserve
alpha::vector< T >, 349
resetSelection
RenderArea, 205
resize
alpha::vector< T >, 349
run
AppDriver, 40

saveShapes
RenderAreaManager, 221
ShapesManager, 265

saveTestimonials
TestimonialManager, 298
saveUsers
UserManager, 338
selection_sort
MainWindow, 121
setA
Ellipse, 62
setAdmin
UserAccount, 326
setAlignment
Text, 316
setB
Ellipse, 62
setBrush
Shape, 250
setCanEdit
ColumnEditDelegate, 54
setDoNotShowAgain
TestimonialManager, 299
setEditPrivileges
RenderArea, 205
setEndPoint
Line, 72
setInternalBrush
Shape, 250
setInternalFont
Text, 316
setInternalPen
Shape, 250
setIsSatisfactory
Testimonial, 282
setLength
Rectangle, 196
Square, 275
Text, 317
setPassword
UserAccount, 326
setPen
Shape, 251
setPointsList
Polygon, 175
Polyline, 184
setR
Circle, 48
setRenderShapes
RenderArea, 205
setSelected
Shape, 251
setShapeSelectedIndex
RenderArea, 205
setShapeType
Shape, 251
setStartPoint
Line, 73
setText
Text, 317
setTrackerId
Shape, 251
setupTestimonials
MainWindow, 122
setUserAccount
UserAccount, 326
setUsername
UserAccount, 326
setWidth
Rectangle, 197
Text, 317
setX
Circle, 49
Ellipse, 62
Line, 73
Polygon, 175
Polyline, 184
Rectangle, 197
Shape, 252
Square, 275
Text, 318
setY
Circle, 49
Ellipse, 63
Line, 74
Polygon, 176
Polyline, 185
Rectangle, 198
Shape, 252
Square, 276
Text, 318
Shape, 223
~Shape, 228
allocateTrackerId, 229
Area, 229
brush, 254
brushItems, 254
childItems, 254
coords, 254
CreateBrushChild, 230
CreateParentItem, 230
CreatePenChild, 232
CreatePointsChild, 232
Draw, 233
getBrush, 233
getBrushColor, 234
getBrushItems, 234
getBrushItemsEnd, 235
getBrushStyle, 235
getChildEnd, 236
getChildItems, 236
getPainter, 237
getParentItem, 238
getPen, 238
getPenCapStyle, 239
getPenColor, 239
getPenItems, 240
getPenItemsEnd, 240
getPenJoinStyle, 241

getPenStyle, 241
getPenWidth, 242
getPoints, 242
getPointsItems, 243
getSelected, 243
getShapeId, 244
getShapeType, 245
getTrackerId, 245
getX, 246
getY, 247
isPointInside, 248
isSelected, 254
Move, 249
nextTracker, 254
operator<, 253
operator=, 249
operator==, 253
painter, 254
parentItem, 255
pen, 255
penItems, 255
Perimeter, 250
pointsItems, 255
setBrush, 250
setInternalBrush, 250
setInternalPen, 250
setPen, 251
setSelected, 251
setShapeType, 251
setTrackerId, 251
setX, 252
setY, 252
Shape, 227, 228
shapeId, 255
shapeType, 255
trackerId, 255
trackersInUse, 256
shape.cpp
operator<, 406
operator==, 406
shape.h
PI, 408
shapeAdded
MainWindow, 123
shapeChanged
MainWindow, 124
shapeDeleted
MainWindow, 124
shapeDimensions
Parser::MorphicShape, 134
shapeId
Parser::MorphicShape, 134
Shape, 255
ShapeIDs
all_shapes.h, 395
shapes
ShapesManager, 267
shapes_to_treeWidget
MainWindow, 125
shapesChanged
ShapesManager, 266
shapeSelectedIndex
RenderArea, 207
ShapesManager, 256
~ShapesManager, 260
addShape, 260
client, 267
deleteAllShapes, 261
deleteShape, 261
getShapesRef, 261
loadShapes, 262
modifyShape, 262
onBadDeleteResponse, 262
onBadGetResponse, 263
onBadPostResponse, 263
onGoodDeleteResponse, 264
onGoodGetResponse, 264
onGoodPostResponse, 265
parse, 267
saveShapes, 265
shapes, 267
shapesChanged, 266
ShapesManager, 259
shapesNotChanged, 266
statusMessage, 267
shapesNotChanged
ShapesManager, 266
ShapesToJson
Parser, 161
shapeTable
MainWindow, 130
shapeType
Parser::MorphicShape, 134
Shape, 255
shouldPromptForTestimonial
TestimonialManager, 300
showLoginPage
LoginWindow, 81
showRenderStatusMessage
MainWindow, 127
showSignupPage
LoginWindow, 81
showTestimonialPrompt
MainWindow, 127
showTestimonialsDisplay
MainWindow, 127
shutdown
AppDriver, 41
signupBtn
LoginWindow, 83
signupPage
LoginWindow, 84
signupPassEdit
LoginWindow, 84
signupRequested
LoginWindow, 82

signupUserEdit
LoginWindow, 84
size
alpha::vector< T >, 349
size_v
alpha::vector< T >, 350
SkipWhitespace
Parser, 162
sortByArea
MainWindow, 128
sortById
MainWindow, 128
sortByPerimeter
MainWindow, 129
sortDropdown
MainWindow, 131
sortOrderDropdown
MainWindow, 131
space
alpha::vector< T >, 350
SQUARE
all_shapes.h, 395
Square, 268
Area, 273
Draw, 273
getLength, 274
isPointInside, 274
length, 277
Perimeter, 275
setLength, 275
setX, 275
setY, 276
Square, 272
stack
LoginWindow, 84
startPoint
Line, 74
startTrackingTime
TestimonialManager, 300
statusMessage
RenderAreaManager, 222
ShapesManager, 267
UserManager, 339
stopTrackingTime
TestimonialManager, 300
StringToVector
Parser, 163

tabWidget
MainWindow, 131
teamNameLabel
MainWindow, 131
Testimonial, 277
fromJson, 280
getAuthor, 281
getContent, 281
getTimestamp, 281
isGuest, 281
isSatisfactory, 281
m_author, 282
m_content, 282
m_isGuest, 282
m_isSatisfactory, 282
m_timestamp, 282
setIsSatisfactory, 282
Testimonial, 280
toJson, 282
TestimonialDialog, 283
m_authorEdit, 287
m_contentEdit, 287
m_doNotShowAgain, 287
onCancel, 286
onSubmit, 286
TestimonialDialog, 285
TestimonialManager, 288
~TestimonialManager, 293
addTestimonial, 294
checkTimeAndPrompt, 294
client, 301
getDoNotShowAgain, 295
getInstance, 295
getSatisfactoryTestimonials, 296
hasUserGivenTestimonial, 297
INITIAL_PROMPT_TIME, 301
loadTestimonials, 297
m_doNotShowAgain, 301
m_testimonials, 301
m_trackingTimer, 301
m_userTimeTracking, 301
onBadGetResponse, 297
onBadPostResponse, 297
onGoodGetResponse, 298
onGoodPostResponse, 298
parse, 301
REPEAT_PROMPT_TIME, 301
saveTestimonials, 298
setDoNotShowAgain, 299
shouldPromptForTestimonial, 300
startTrackingTime, 300
stopTrackingTime, 300
TestimonialManager, 293
TestimonialsDisplayDialog, 302
m_testimonialsLayout, 305
refreshTestimonials, 304
TestimonialsDisplayDialog, 303
TestimonialsToJson
Parser, 163
TEXT
all_shapes.h, 395
Text, 305
Area, 311
Draw, 311
font, 319
getFont, 312
getFontStyle, 312
getFontWeight, 313
getLength, 313

getTextAlignment, 314
getTextColor, 314
getTextString, 314
getWidth, 315
isPointInside, 315
length, 319
Perimeter, 316
setAlignment, 316
setInternalFont, 316
setLength, 317
setText, 317
setWidth, 317
setX, 318
setY, 318
Text, 310
textAlignment, 319
textColor, 319
textString, 320
width, 320
textAlignment
Parser::MorphicShape, 134
Text, 319
textColor
Parser::MorphicShape, 134
Text, 319
textString
Parser::MorphicShape, 134
Text, 320
toJson
Testimonial, 282
toSignupBtn
LoginWindow, 84
trackerId
Parser::MorphicShape, 134
Shape, 255
trackersInUse
Shape, 256

Ui, 13
ui
MainWindow, 131
UpdateAccumulator
Parser, 164
updateShapeDisplayCoords
RenderArea, 205
UpdateUserAccumulator
Parser, 165
user
AppDriver, 41
UserAccount, 320
∼UserAccount, 323
admin, 327
getPassword, 324
getUsername, 324
isAdmin, 325
operator=, 325
password, 327
setAdmin, 326
setPassword, 326
setUserAccount, 326
setUsername, 326
UserAccount, 323, 324
username, 327
userAuthenticated
UserManager, 339
userChanged
UserManager, 340
UserManager, 327
∼UserManager, 332
addUser, 332
authenticate, 333
authenticationFailed, 333
client, 340
currUser, 340
deleteAllUsers, 334
deleteUser, 334
getCurrUserRef, 334
loadUsers, 334
modifyUser, 335
onBadDeleteResponse, 335
onBadGetResponse, 336
onBadPostResponse, 336
onGoodDeleteResponse, 337
onGoodGetResponse, 337
onGoodPostResponse, 338
parse, 341
saveUsers, 338
statusMessage, 339
userAuthenticated, 339
userChanged, 340
UserManager, 331
userNotChanged, 340
users, 341
username
LoginWindow, 82
Parser::RawUser, 189
UserAccount, 327
userNotChanged
UserManager, 340
users
UserManager, 341
userStatusLabel
MainWindow, 131
UsersToJson
Parser, 165

vector
alpha::vector< T >, 344, 345
vector.h
ALPHA_VECTOR_H, 414

webservice.cpp
get_json_file, 420
main, 420
width
Rectangle, 198
Text, 320