# Custom IRC

Generated by Doxygen 1.8.3.1

Wed Apr 10 2013 01:58:44

# Contents


**1 Todo List** **1**

**2 Namespace Index** **3**

2.1 Namespace List . . . . . 3

**3 Hierarchical Index** **5**

3.1 Class Hierarchy . . . . . 5

**4 Class Index** **7**

4.1 Class List . . . . . 7

**5 File Index** **9**

5.1 File List . . . . . 9

**6 Namespace Documentation** **11**

6.1 client Namespace Reference . . . . . 11

6.2 ircCodes Namespace Reference . . . . . 11

6.2.1 Enumeration Type Documentation . . . . . 15

6.2.1.1 ircCode . . . . . 15

6.3 K4U Namespace Reference . . . . . 25

6.3.1 Detailed Description . . . . . 25

6.3.2 Enumeration Type Documentation . . . . . 25

6.3.2.1 jsonCommands . . . . . 25

6.3.2.2 logTags . . . . . 26

6.3.3 Function Documentation . . . . . 26

6.3.3.1 endl . . . . . 26

6.3.3.2 operator<< . . . . . 27

6.3.3.3 operator<< . . . . . 28

6.4 Package org . . . . . 28

6.5 Package org.k4u . . . . . 28

6.6 Package org.k4u.customIrcClient . . . . . 28

6.6.1 Detailed Description . . . . . 29

6.7 Package org.kde . . . . . 29

6.8 Package org.kde.necessitas . . . 29
6.9 Package org.kde.necessitas.ministro . . . 29
6.10 Package org.kde.necessitas.origo . . . 29
6.11 QtJson Namespace Reference . . . 29
6.11.1 Detailed Description . . . 30
6.11.2 Enumeration Type Documentation . . . 30
6.11.2.1 JsonToken . . . 30
6.11.3 Function Documentation . . . 30
6.11.3.1 parse . . . 30
6.11.3.2 parse . . . 31
6.11.3.3 serialize . . . 31
6.11.3.4 serialize . . . 31
6.12 server Namespace Reference . . . 32
6.12.1 Function Documentation . . . 33
6.12.1.1 getColor . . . 33
6.12.1.2 getId . . . 33
6.12.1.3 getNickAndStatus . . . 33
6.12.1.4 getNickAndStatusFromId . . . 34
6.12.1.5 inCommand . . . 34
6.12.1.6 operator<< . . . 35
6.12.1.7 qts . . . 35
6.13 shared Namespace Reference . . . 36
6.13.1 Enumeration Type Documentation . . . 36
6.13.1.1 level . . . 36
6.14 shared::events Namespace Reference . . . 36
6.14.1 Enumeration Type Documentation . . . 37
6.14.1.1 eventTypes . . . 37
6.15 Ui Namespace Reference . . . 37

**7 Class Documentation** **39**
7.1 org.k4u.customIrcClient.BuildConfig Class Reference . . . 39
7.1.1 Detailed Description . . . 39
7.1.2 Member Data Documentation . . . 39
7.1.2.1 DEBUG . . . 39
7.2 client::chatBox Class Reference . . . 40
7.2.1 Detailed Description . . . 42
7.2.2 Member Enumeration Documentation . . . 42
7.2.2.1 widgetTypes . . . 42
7.2.3 Constructor & Destructor Documentation . . . 42
7.2.3.1 chatBox . . . 42

7.2.3.2 ~chatBox . . . 42
7.2.4 Member Function Documentation . . . 42
7.2.4.1 addChat . . . 42
7.2.4.2 addChat . . . 43
7.2.4.3 addMessage . . . 43
7.2.4.4 addMessage . . . 43
7.2.4.5 addMessage . . . 44
7.2.4.6 addMessage . . . 44
7.2.4.7 addMessage . . . 45
7.2.4.8 setAllUserNameSize . . . 45
7.2.5 Member Data Documentation . . . 45
7.2.5.1 entries . . . 45
7.2.5.2 ui . . . 45
7.2.5.3 widthOfNick . . . 46
7.3 client::chatEntry Class Reference . . . 46
7.3.1 Detailed Description . . . 48
7.3.2 Constructor & Destructor Documentation . . . 48
7.3.2.1 chatEntry . . . 48
7.3.2.2 chatEntry . . . 49
7.3.2.3 ~chatEntry . . . 49
7.3.3 Member Function Documentation . . . 49
7.3.3.1 calculateMessageWidth . . . 49
7.3.3.2 getHeight . . . 50
7.3.3.3 getWidth . . . 50
7.3.3.4 init . . . 51
7.3.3.5 resizeEvent . . . 51
7.3.3.6 setUserWidth . . . 51
7.3.4 Member Data Documentation . . . 52
7.3.4.1 message . . . 52
7.3.4.2 name . . . 52
7.3.4.3 timeOfMessage . . . 52
7.3.4.4 ui . . . 52
7.4 client::chatInput Class Reference . . . 53
7.4.1 Detailed Description . . . 54
7.4.2 Constructor & Destructor Documentation . . . 54
7.4.2.1 chatInput . . . 54
7.4.3 Member Function Documentation . . . 54
7.4.3.1 checkForCommand . . . 54
7.4.3.2 commandGiven . . . 55
7.4.3.3 keyPressEvent . . . 55

7.4.3.4 sendChat . . . 56
7.5 client::chatNotification Class Reference . . . 56
7.5.1 Detailed Description . . . 59
7.5.2 Constructor & Destructor Documentation . . . 59
7.5.2.1 chatNotification . . . 59
7.5.2.2 chatNotification . . . 59
7.5.2.3 ∼chatNotification . . . 60
7.5.3 Member Function Documentation . . . 60
7.5.3.1 calculateMessageWidth . . . 60
7.5.3.2 getHeight . . . 61
7.5.3.3 getWidth . . . 61
7.5.3.4 resizeEvent . . . 61
7.5.4 Member Data Documentation . . . 62
7.5.4.1 message . . . 62
7.5.4.2 timeOfMessage . . . 62
7.5.4.3 ui . . . 62
7.6 shared::events::clsEvent Class Reference . . . 62
7.6.1 Detailed Description . . . 64
7.6.2 Constructor & Destructor Documentation . . . 64
7.6.2.1 clsEvent . . . 64
7.6.2.2 clsEvent . . . 64
7.6.3 Member Function Documentation . . . 65
7.6.3.1 addToData . . . 65
7.6.3.2 getData . . . 66
7.6.3.3 getEventType . . . 66
7.6.3.4 getTime . . . 67
7.6.3.5 getType . . . 67
7.6.3.6 setEventType . . . 67
7.6.3.7 toVariant . . . 68
7.6.4 Member Data Documentation . . . 68
7.6.4.1 data . . . 69
7.6.4.2 log . . . 69
7.6.4.3 ourType . . . 69
7.6.4.4 timeOfEvent . . . 69
7.7 K4U::clsLog Class Reference . . . 69
7.7.1 Detailed Description . . . 70
7.7.2 Constructor & Destructor Documentation . . . 70
7.7.2.1 clsLog . . . 70
7.7.3 Member Function Documentation . . . 71
7.7.3.1 doEndl . . . 71

7.7.3.2 getTag . . . 72
7.7.3.3 operator<< . . . 73
7.7.3.4 operator<< . . . 73
7.7.3.5 operator<< . . . 73
7.7.3.6 operator<< . . . 73
7.7.3.7 operator<< . . . 73
7.7.3.8 operator<< . . . 74
7.7.3.9 operator<< . . . 74
7.7.3.10 operator<< . . . 74
7.7.3.11 operator<< . . . 74
7.7.3.12 operator<< . . . 74
7.7.3.13 operator<< . . . 74
7.7.3.14 operator<< . . . 75
7.7.3.15 operator<< . . . 75
7.7.3.16 prefix . . . 75
7.7.3.17 setPrefix . . . 75
7.7.4 Member Data Documentation . . . 76
7.7.4.1 isFirst . . . 76
7.7.4.2 prf . . . 76
7.7.4.3 tag . . . 76
7.8 client::clsNotify Class Reference . . . 76
7.8.1 Detailed Description . . . 77
7.8.2 Constructor & Destructor Documentation . . . 78
7.8.2.1 clsNotify . . . 78
7.8.3 Member Function Documentation . . . 78
7.8.3.1 show . . . 78
7.9 client::clsServerConn Class Reference . . . 78
7.9.1 Detailed Description . . . 82
7.9.2 Constructor & Destructor Documentation . . . 82
7.9.2.1 clsServerConn . . . 82
7.9.3 Member Function Documentation . . . 83
7.9.3.1 chatReceived . . . 83
7.9.3.2 checkForJsonCommand . . . 83
7.9.3.3 connected . . . 84
7.9.3.4 connectTimeout . . . 84
7.9.3.5 disconnected . . . 85
7.9.3.6 doConnect . . . 85
7.9.3.7 handleCompletedQuery . . . 86
7.9.3.8 handleEventReceived . . . 86
7.9.3.9 handleEventRequest . . . 87

7.9.3.10 handleOwnUserChange . . . 88
7.9.3.11 readData . . . 89
7.9.3.12 sendCommand . . . 90
7.9.3.13 serverMessageReceived . . . 91
7.9.3.14 sgnConnectTimeout . . . 91
7.9.3.15 srvConnected . . . 91
7.9.3.16 srvDisconnected . . . 92
7.9.3.17 userChangeNick . . . 92
7.9.3.18 userEnter . . . 93
7.9.3.19 userLeave . . . 93
7.9.3.20 userQueryCompleted . . . 93
7.9.3.21 userStatusChange . . . 94
7.9.4 Member Data Documentation . . . 94
7.9.4.1 buffer . . . 94
7.9.4.2 connectTimer . . . 94
7.9.4.3 depth . . . 94
7.9.4.4 hostName . . . 94
7.9.4.5 isConnected . . . 94
7.9.4.6 isConnecting . . . 95
7.9.4.7 log . . . 95
7.9.4.8 port . . . 95
7.9.4.9 sock . . . 95
7.10 client::clsSettings Class Reference . . . 95
7.10.1 Detailed Description . . . 96
7.10.2 Constructor & Destructor Documentation . . . 96
7.10.2.1 clsSettings . . . 96
7.10.3 Member Function Documentation . . . 96
7.10.3.1 getPort . . . 96
7.10.3.2 getServer . . . 97
7.10.3.3 readServer . . . 97
7.10.4 Member Data Documentation . . . 97
7.10.4.1 port . . . 98
7.10.4.2 server . . . 98
7.11 server::clsSettings Class Reference . . . 98
7.11.1 Detailed Description . . . 100
7.11.2 Constructor & Destructor Documentation . . . 100
7.11.2.1 clsSettings . . . 100
7.11.3 Member Function Documentation . . . 101
7.11.3.1 getChannel . . . 101
7.11.3.2 getListeningPort . . . 101

7.11.3.3 getOwnUser . . . 102
7.11.3.4 getServer . . . 102
7.11.3.5 getUserName . . . 103
7.11.3.6 getUsers . . . 103
7.11.3.7 readServer . . . 104
7.11.3.8 readUiSettings . . . 104
7.11.3.9 readUserInfo . . . 105
7.11.3.10 readUsers . . . 105
7.11.4 Member Data Documentation . . . 106
7.11.4.1 channel . . . 106
7.11.4.2 listeningPort . . . 106
7.11.4.3 ownUser . . . 106
7.11.4.4 server . . . 106
7.11.4.5 users . . . 107
7.12 client::clsUserEntry Class Reference . . . 107
7.12.1 Detailed Description . . . 109
7.12.2 Constructor & Destructor Documentation . . . 109
7.12.2.1 clsUserEntry . . . 109
7.12.2.2 ∼clsUserEntry . . . 109
7.12.3 Member Function Documentation . . . 109
7.12.3.1 iconChanged . . . 109
7.12.3.2 refresh . . . 110
7.12.4 Member Data Documentation . . . 110
7.12.4.1 ui . . . 111
7.12.4.2 user . . . 111
7.13 client::clsUserList Class Reference . . . 111
7.13.1 Detailed Description . . . 113
7.13.2 Constructor & Destructor Documentation . . . 113
7.13.2.1 clsUserList . . . 113
7.13.2.2 ∼clsUserList . . . 113
7.13.3 Member Function Documentation . . . 113
7.13.3.1 addUser . . . 113
7.13.3.2 clear . . . 113
7.13.3.3 refreshUser . . . 114
7.13.3.4 removeUser . . . 114
7.13.3.5 setupUi . . . 114
7.13.4 Member Data Documentation . . . 115
7.13.4.1 users . . . 115
7.13.4.2 verticalLayout . . . 115
7.13.4.3 verticalSpacer . . . 115

7.14 client::chatBox::content Struct Reference . . . 115
7.14.1 Detailed Description . . . 115
7.14.2 Member Data Documentation . . . 115
7.14.2.1 type . . . 115
7.14.2.2 widget . . . 116
7.15 shared::events::eventChat Class Reference . . . 116
7.15.1 Detailed Description . . . 118
7.15.2 Constructor & Destructor Documentation . . . 118
7.15.2.1 eventChat . . . 118
7.15.2.2 eventChat . . . 118
7.15.3 Member Function Documentation . . . 118
7.15.3.1 getMessage . . . 119
7.15.3.2 getUser . . . 119
7.16 shared::events::eventServerMessage Class Reference . . . 120
7.16.1 Detailed Description . . . 122
7.16.2 Constructor & Destructor Documentation . . . 122
7.16.2.1 eventServerMessage . . . 122
7.16.2.2 eventServerMessage . . . 122
7.16.3 Member Function Documentation . . . 122
7.16.3.1 getLevel . . . 123
7.16.3.2 getMessage . . . 123
7.17 shared::events::eventUserChangeNick Class Reference . . . 124
7.17.1 Detailed Description . . . 127
7.17.2 Constructor & Destructor Documentation . . . 127
7.17.2.1 eventUserChangeNick . . . 127
7.17.2.2 eventUserChangeNick . . . 127
7.17.3 Member Function Documentation . . . 128
7.17.3.1 getNewNick . . . 128
7.17.3.2 getOldNick . . . 128
7.17.3.3 getUserId . . . 128
7.18 shared::events::eventUserChangeStatus Class Reference . . . 129
7.18.1 Detailed Description . . . 132
7.18.2 Constructor & Destructor Documentation . . . 132
7.18.2.1 eventUserChangeStatus . . . 132
7.18.2.2 eventUserChangeStatus . . . 132
7.18.3 Member Function Documentation . . . 133
7.18.3.1 getNewStatus . . . 133
7.18.3.2 getOldStatus . . . 133
7.18.3.3 getUserId . . . 133
7.19 shared::events::eventUserJoin Class Reference . . . 134

7.19.1 Detailed Description . . . 136
7.19.2 Constructor & Destructor Documentation . . . 136
7.19.2.1 eventUserJoin . . . 136
7.19.2.2 eventUserJoin . . . 136
7.19.3 Member Function Documentation . . . 136
7.19.3.1 getUser . . . 137
7.19.3.2 getUserId . . . 137
7.20 shared::events::eventUserLeave Class Reference . . . 138
7.20.1 Detailed Description . . . 140
7.20.2 Constructor & Destructor Documentation . . . 140
7.20.2.1 eventUserLeave . . . 140
7.20.2.2 eventUserLeave . . . 140
7.20.3 Member Function Documentation . . . 140
7.20.3.1 getUserId . . . 140
7.21 org.kde.necessitas.ministro.IMinistro Interface Reference . . . 142
7.21.1 Detailed Description . . . 143
7.21.2 Member Function Documentation . . . 143
7.21.2.1 requestLoader . . . 143
7.22 org.kde.necessitas.ministro.IMinistroCallback Interface Reference . . . 144
7.22.1 Detailed Description . . . 145
7.22.2 Member Function Documentation . . . 145
7.22.2.1 loaderReady . . . 145
7.23 server::ircClient Class Reference . . . 146
7.23.1 Detailed Description . . . 150
7.23.2 Constructor & Destructor Documentation . . . 151
7.23.2.1 ircClient . . . 151
7.23.2.2 ~ircClient . . . 151
7.23.3 Member Function Documentation . . . 151
7.23.3.1 chatReceived . . . 151
7.23.3.2 checkForPing . . . 152
7.23.3.3 connectTimeout . . . 152
7.23.3.4 disconnected . . . 153
7.23.3.5 doConnect . . . 154
7.23.3.6 getOwnNick . . . 154
7.23.3.7 handleCommand . . . 154
7.23.3.8 init . . . 155
7.23.3.9 IRCConnected . . . 156
7.23.3.10 IRCConnectTimeOut . . . 157
7.23.3.11 IRCDisconnected . . . 157
7.23.3.12 readData . . . 157

7.23.3.13 send . . . 158
7.23.3.14 sendChat . . . 159
7.23.3.15 sendChat . . . 160
7.23.3.16 sendChat . . . 160
7.23.3.17 sendPong . . . 161
7.23.3.18 setOwnNick . . . 161
7.23.3.19 setOwnUser . . . 161
7.23.3.20 setUser . . . 162
7.23.3.21 sltConnected . . . 162
7.23.3.22 startupComplete . . . 162
7.23.3.23 userChangeNick . . . 163
7.23.3.24 userChangeStatus . . . 163
7.23.3.25 userOffline . . . 163
7.23.3.26 userOnline . . . 164
7.23.4 Member Data Documentation . . . 164
7.23.4.1 buffer . . . 164
7.23.4.2 channel . . . 164
7.23.4.3 connected . . . 164
7.23.4.4 connecting . . . 164
7.23.4.5 connectTimer . . . 165
7.23.4.6 hostname . . . 165
7.23.4.7 lastPing . . . 165
7.23.4.8 log . . . 165
7.23.4.9 motdReceived . . . 165
7.23.4.10 ownNickAndStatus . . . 165
7.23.4.11 ownUser . . . 165
7.23.4.12 port . . . 165
7.23.4.13 sock . . . 165
7.23.4.14 suppressDisconnectSignal . . . 165
7.23.4.15 timeBetweenPings . . . 165
7.23.4.16 timeOutChecker . . . 166
7.24 server::ircUser Class Reference . . . 166
7.24.1 Detailed Description . . . 168
7.24.2 Constructor & Destructor Documentation . . . 169
7.24.2.1 ircUser . . . 169
7.24.2.2 ircUser . . . 169
7.24.2.3 ircUser . . . 169
7.24.2.4 ircUser . . . 169
7.24.3 Member Function Documentation . . . 169
7.24.3.1 getId . . . 169

7.24.3.2 getIrcId . . . 170
7.24.3.3 getName . . . 170
7.24.3.4 getOnline . . . 171
7.24.3.5 getStatus . . . 171
7.24.3.6 isStandard . . . 171
7.24.3.7 operator= . . . 171
7.24.3.8 setColor . . . 171
7.24.3.9 setColor . . . 171
7.24.3.10 setEmail . . . 172
7.24.3.11 setId . . . 172
7.24.3.12 setIrcId . . . 172
7.24.3.13 setNick . . . 172
7.24.3.14 setOnline . . . 172
7.24.3.15 setStatus . . . 173
7.24.3.16 toVariantMap . . . 173
7.24.4 Friends And Related Function Documentation . . . 174
7.24.4.1 operator<< . . . 174
7.24.5 Member Data Documentation . . . 174
7.24.5.1 email . . . 174
7.24.5.2 id . . . 174
7.24.5.3 ircId . . . 174
7.24.5.4 log . . . 174
7.24.5.5 name . . . 174
7.24.5.6 online . . . 174
7.24.5.7 standard . . . 174
7.24.5.8 status . . . 174
7.24.5.9 uColor . . . 174
7.25 client::ircUser Class Reference . . . 175
7.25.1 Detailed Description . . . 177
7.25.2 Constructor & Destructor Documentation . . . 177
7.25.2.1 ircUser . . . 177
7.25.2.2 ircUser . . . 178
7.25.2.3 ~ircUser . . . 178
7.25.3 Member Function Documentation . . . 178
7.25.3.1 copyFrom . . . 178
7.25.3.2 getColor . . . 178
7.25.3.3 getEmail . . . 179
7.25.3.4 getIcon . . . 179
7.25.3.5 getId . . . 179
7.25.3.6 getNick . . . 179

7.25.3.7 getOnline . . . 180
7.25.3.8 getStandard . . . 180
7.25.3.9 getStatus . . . 180
7.25.3.10 iconChanged . . . 181
7.25.3.11 setColor . . . 181
7.25.3.12 setEmail . . . 181
7.25.3.13 setNick . . . 181
7.25.3.14 setOnline . . . 181
7.25.3.15 setStatus . . . 181
7.25.3.16 slot_netwManagerFinished . . . 181
7.25.4 Member Data Documentation . . . 182
7.25.4.1 email . . . 182
7.25.4.2 icon . . . 182
7.25.4.3 id . . . 182
7.25.4.4 log . . . 182
7.25.4.5 nick . . . 182
7.25.4.6 online . . . 182
7.25.4.7 standard . . . 183
7.25.4.8 status . . . 183
7.25.4.9 uColor . . . 183
7.26 server::ircUserList Class Reference . . . 183
7.26.1 Detailed Description . . . 184
7.26.2 Constructor & Destructor Documentation . . . 184
7.26.2.1 ircUserList . . . 184
7.26.3 Member Function Documentation . . . 184
7.26.3.1 add . . . 184
7.26.3.2 del . . . 185
7.26.3.3 findUser . . . 186
7.26.3.4 getAll . . . 186
7.26.3.5 getUser . . . 186
7.26.3.6 getUserById . . . 187
7.26.3.7 getUserByIrcId . . . 188
7.26.3.8 getUserByNick . . . 188
7.26.4 Member Data Documentation . . . 189
7.26.4.1 lastId . . . 189
7.26.4.2 users . . . 189
7.27 K4U::jsonCommand Class Reference . . . 189
7.27.1 Detailed Description . . . 190
7.27.2 Constructor & Destructor Documentation . . . 190
7.27.2.1 jsonCommand . . . 190

7.27.2.2 jsonCommand . . . 191
7.27.2.3 jsonCommand . . . 191
7.27.3 Member Function Documentation . . . 191
7.27.3.1 addToData . . . 191
7.27.3.2 getCommand . . . 192
7.27.3.3 getCommandString . . . 192
7.27.3.4 getData . . . 193
7.27.3.5 getData . . . 194
7.27.3.6 operator= . . . 194
7.27.3.7 setCommand . . . 194
7.27.3.8 setCommandInt . . . 194
7.27.3.9 setData . . . 195
7.27.3.10 toJsonString . . . 195
7.27.4 Friends And Related Function Documentation . . . 196
7.27.4.1 operator<< . . . 196
7.27.4.2 operator<< . . . 196
7.27.5 Member Data Documentation . . . 196
7.27.5.1 command . . . 196
7.27.5.2 data . . . 196
7.28 server::mainApp Class Reference . . . 196
7.28.1 Detailed Description . . . 200
7.28.2 Constructor & Destructor Documentation . . . 200
7.28.2.1 mainApp . . . 200
7.28.3 Member Function Documentation . . . 200
7.28.3.1 addToEventHistory . . . 200
7.28.3.2 chatReceived . . . 200
7.28.3.3 doIrcReconnect . . . 201
7.28.3.4 doSendEvents . . . 202
7.28.3.5 doUserQuery . . . 202
7.28.3.6 ircConnected . . . 203
7.28.3.7 ircConnectTimeOut . . . 203
7.28.3.8 ircDisconnected . . . 204
7.28.3.9 ownUserChangeStatus . . . 204
7.28.3.10 run . . . 205
7.28.3.11 serverMessage . . . 206
7.28.3.12 uiSendChat . . . 207
7.28.3.13 userChangeNick . . . 208
7.28.3.14 userChangeStatus . . . 208
7.28.3.15 userOffline . . . 209
7.28.3.16 userOnline . . . 210

7.28.4 Member Data Documentation . . . 211
7.28.4.1 events . . . 211
7.28.4.2 irc . . . 211
7.28.4.3 log . . . 211
7.28.4.4 ownUser . . . 212
7.28.4.5 reconnectTimer . . . 212
7.28.4.6 settings . . . 212
7.28.4.7 TAG . . . 212
7.28.4.8 ui . . . 212
7.28.4.9 users . . . 212
7.29 MainWindow Class Reference . . . 212
7.29.1 Detailed Description . . . 214
7.29.2 Member Enumeration Documentation . . . 214
7.29.2.1 ScreenOrientation . . . 214
7.29.3 Constructor & Destructor Documentation . . . 214
7.29.3.1 MainWindow . . . 214
7.29.3.2 ~MainWindow . . . 214
7.29.4 Member Function Documentation . . . 214
7.29.4.1 setOrientation . . . 214
7.29.4.2 showExpanded . . . 215
7.29.5 Member Data Documentation . . . 215
7.29.5.1 ui . . . 215
7.30 client::MainWindow Class Reference . . . 215
7.30.1 Detailed Description . . . 218
7.30.2 Constructor & Destructor Documentation . . . 218
7.30.2.1 MainWindow . . . 218
7.30.2.2 ~MainWindow . . . 219
7.30.3 Member Function Documentation . . . 219
7.30.3.1 chatReceived . . . 219
7.30.3.2 commandGiven . . . 220
7.30.3.3 connectedTimeout . . . 221
7.30.3.4 doReconnect . . . 221
7.30.3.5 moveScrollBarToBottom . . . 222
7.30.3.6 sendChat . . . 222
7.30.3.7 serverConnected . . . 223
7.30.3.8 serverDisconnected . . . 224
7.30.3.9 serverMessageReceived . . . 224
7.30.3.10 showMessage . . . 225
7.30.3.11 userChangeNick . . . 225
7.30.3.12 userEnter . . . 226

7.30.3.13 userLeave . . . 227
7.30.3.14 userQueryCompleted . . . 228
7.30.3.15 userStatusChange . . . 228
7.30.4 Member Data Documentation . . . 229
7.30.4.1 cb . . . 229
7.30.4.2 log . . . 229
7.30.4.3 reconnectTimer . . . 229
7.30.4.4 server . . . 229
7.30.4.5 settings . . . 230
7.30.4.6 ui . . . 230
7.30.4.7 users . . . 230
7.31 server::nickAndStatus Struct Reference . . . 230
7.31.1 Detailed Description . . . 230
7.31.2 Member Data Documentation . . . 230
7.31.2.1 nick . . . 231
7.31.2.2 status . . . 231
7.32 org.kde.necessitas.origo.QtActivity Class Reference . . . 232
7.32.1 Detailed Description . . . 236
7.32.2 Member Function Documentation . . . 236
7.32.2.1 dispatchKeyEvent . . . 236
7.32.2.2 dispatchPopulateAccessibilityEvent . . . 237
7.32.2.3 dispatchTouchEvent . . . 237
7.32.2.4 dispatchTrackballEvent . . . 237
7.32.2.5 downloadUpgradeMinistro . . . 237
7.32.2.6 loadApplication . . . 238
7.32.2.7 ministroNotFound . . . 238
7.32.2.8 onActivityResult . . . 239
7.32.2.9 onApplyThemeResource . . . 239
7.32.2.10 onAttachedToWindow . . . 239
7.32.2.11 onBackPressed . . . 240
7.32.2.12 onChildTitleChanged . . . 240
7.32.2.13 onConfigurationChanged . . . 240
7.32.2.14 onContentChanged . . . 240
7.32.2.15 onContextItemSelected . . . 241
7.32.2.16 onContextMenuClosed . . . 241
7.32.2.17 onCreate . . . 241
7.32.2.18 onCreateContextMenu . . . 241
7.32.2.19 onCreateDescription . . . 242
7.32.2.20 onCreateDialog . . . 242
7.32.2.21 onCreateDialog . . . 242

7.32.2.22 onCreateOptionsMenu . . . 242
7.32.2.23 onCreatePanelMenu . . . 243
7.32.2.24 onCreatePanelView . . . 243
7.32.2.25 onCreateThumbnail . . . 243
7.32.2.26 onCreateView . . . 243
7.32.2.27 onDestroy . . . 244
7.32.2.28 onDetachedFromWindow . . . 244
7.32.2.29 onKeyDown . . . 244
7.32.2.30 onKeyLongPress . . . 244
7.32.2.31 onKeyMultiple . . . 245
7.32.2.32 onKeyUp . . . 245
7.32.2.33 onLowMemory . . . 245
7.32.2.34 onMenuItemSelected . . . 246
7.32.2.35 onMenuOpened . . . 246
7.32.2.36 onNewIntent . . . 246
7.32.2.37 onOptionsItemSelected . . . 246
7.32.2.38 onOptionsMenuClosed . . . 247
7.32.2.39 onPanelClosed . . . 247
7.32.2.40 onPause . . . 247
7.32.2.41 onPostCreate . . . 247
7.32.2.42 onPostResume . . . 248
7.32.2.43 onPrepareDialog . . . 248
7.32.2.44 onPrepareDialog . . . 248
7.32.2.45 onPrepareOptionsMenu . . . 248
7.32.2.46 onPreparePanel . . . 249
7.32.2.47 onRestart . . . 249
7.32.2.48 onRestoreInstanceState . . . 249
7.32.2.49 onResume . . . 249
7.32.2.50 onRetainNonConfigurationInstance . . . 250
7.32.2.51 onSaveInstanceState . . . 250
7.32.2.52 onSearchRequested . . . 250
7.32.2.53 onStart . . . 250
7.32.2.54 onStop . . . 251
7.32.2.55 onTitleChanged . . . 251
7.32.2.56 onTouchEvent . . . 251
7.32.2.57 onTrackballEvent . . . 252
7.32.2.58 onUserInteraction . . . 252
7.32.2.59 onUserLeaveHint . . . 252
7.32.2.60 onWindowAttributesChanged . . . 252
7.32.2.61 onWindowFocusChanged . . . 253

7.32.2.62 startApp . . . 253
7.32.2.63 super_dispatchKeyEvent . . . 253
7.32.2.64 super_dispatchPopulateAccessibilityEvent . . . 253
7.32.2.65 super_dispatchTouchEvent . . . 254
7.32.2.66 super_dispatchTrackballEvent . . . 254
7.32.2.67 super_onActivityResult . . . 254
7.32.2.68 super_onApplyThemeResource . . . 254
7.32.2.69 super_onAttachedToWindow . . . 254
7.32.2.70 super_onBackPressed . . . 254
7.32.2.71 super_onChildTitleChanged . . . 254
7.32.2.72 super_onConfigurationChanged . . . 254
7.32.2.73 super_onContentChanged . . . 254
7.32.2.74 super_onContextItemSelected . . . 254
7.32.2.75 super_onContextMenuClosed . . . 254
7.32.2.76 super_onCreateContextMenu . . . 255
7.32.2.77 super_onCreateDescription . . . 255
7.32.2.78 super_onCreateDialog . . . 255
7.32.2.79 super_onCreateDialog . . . 255
7.32.2.80 super_onCreateOptionsMenu . . . 255
7.32.2.81 super_onCreatePanelMenu . . . 255
7.32.2.82 super_onCreatePanelView . . . 255
7.32.2.83 super_onCreateThumbnail . . . 255
7.32.2.84 super_onCreateView . . . 255
7.32.2.85 super_onDetachedFromWindow . . . 255
7.32.2.86 super_onKeyDown . . . 255
7.32.2.87 super_onKeyLongPress . . . 255
7.32.2.88 super_onKeyMultiple . . . 256
7.32.2.89 super_onKeyUp . . . 256
7.32.2.90 super_onMenuItemSelected . . . 256
7.32.2.91 super_onMenuOpened . . . 256
7.32.2.92 super_onNewIntent . . . 256
7.32.2.93 super_onOptionsItemSelected . . . 256
7.32.2.94 super_onOptionsMenuClosed . . . 256
7.32.2.95 super_onPanelClosed . . . 256
7.32.2.96 super_onPrepareDialog . . . 256
7.32.2.97 super_onPrepareDialog . . . 256
7.32.2.98 super_onPrepareOptionsMenu . . . 256
7.32.2.99 super_onPreparePanel . . . 256
7.32.2.100 super_onRestoreInstanceState . . . 257
7.32.2.101 super_onRetainNonConfigurationInstance . . . 257

7.32.2.102 super_onSaveInstanceState . . . 257
7.32.2.103 super_onSearchRequested . . . 257
7.32.2.104 super_onTitleChanged . . . 257
7.32.2.105 super_onTouchEvent . . . 257
7.32.2.106 super_onTrackballEvent . . . 257
7.32.2.107 super_onUserInteraction . . . 257
7.32.2.108 super_onUserLeaveHint . . . 257
7.32.2.109 super_onWindowAttributesChanged . . . 257
7.32.2.110 super_onWindowFocusChanged . . . 257
7.32.3 Member Data Documentation . . . 258
7.32.3.1 APPLICATION_PARAMETERS . . . 258
7.32.3.2 APPLICATION_PARAMETERS_KEY . . . 258
7.32.3.3 APPLICATION_TITLE_KEY . . . 258
7.32.3.4 BUNDLED_LIBRARIES_KEY . . . 258
7.32.3.5 DEX_PATH_KEY . . . 258
7.32.3.6 ENVIRONMENT_VARIABLES . . . 258
7.32.3.7 ENVIRONMENT_VARIABLES_KEY . . . 258
7.32.3.8 ERROR_CODE_KEY . . . 258
7.32.3.9 ERROR_MESSAGE_KEY . . . 258
7.32.3.10 INCOMPATIBLE_MINISTRO_VERSION . . . 258
7.32.3.11 LIB_PATH_KEY . . . 258
7.32.3.12 LOADER_CLASS_NAME_KEY . . . 259
7.32.3.13 m_activityInfo . . . 259
7.32.3.14 m_classLoader . . . 259
7.32.3.15 m_ministroConnection . . . 259
7.32.3.16 m_qtLibs . . . 259
7.32.3.17 MAIN_LIBRARY_KEY . . . 259
7.32.3.18 MINIMUM_MINISTRO_API_KEY . . . 259
7.32.3.19 MINIMUM_QT_VERSION_KEY . . . 259
7.32.3.20 MINISTRO_API_LEVEL . . . 259
7.32.3.21 MINISTRO_INSTALL_REQUEST_CODE . . . 259
7.32.3.22 NATIVE_LIBRARIES_KEY . . . 259
7.32.3.23 NECESSITAS_API_LEVEL . . . 260
7.32.3.24 NECESSITAS_API_LEVEL_KEY . . . 260
7.32.3.25 QT_PROVIDER . . . 260
7.32.3.26 QT_PROVIDER_KEY . . . 260
7.32.3.27 QT_VERSION . . . 260
7.32.3.28 REQUIRED_MODULES_KEY . . . 260
7.33 org.kde.necessitas.origo.QtApplication Class Reference . . . 261
7.33.1 Detailed Description . . . 263

7.33.2 Member Function Documentation . . . 263
7.33.2.1 invokeDelegate . . . 263
7.33.2.2 invokeDelegateMethod . . . 264
7.33.2.3 onTerminate . . . 265
7.33.2.4 setQtActivityDelegate . . . 266
7.33.3 Member Data Documentation . . . 266
7.33.3.1 dispatchGenericMotionEvent . . . 266
7.33.3.2 dispatchKeyEvent . . . 266
7.33.3.3 dispatchKeyShortcutEvent . . . 266
7.33.3.4 dispatchPopulateAccessibilityEvent . . . 266
7.33.3.5 dispatchTouchEvent . . . 266
7.33.3.6 dispatchTrackballEvent . . . 266
7.33.3.7 m_delegateMethods . . . 267
7.33.3.8 m_delegateObject . . . 267
7.33.3.9 onActivityResult . . . 267
7.33.3.10 onCreate . . . 267
7.33.3.11 onGenericMotionEvent . . . 267
7.33.3.12 onKeyDown . . . 267
7.33.3.13 onKeyLongPress . . . 267
7.33.3.14 onKeyMultiple . . . 267
7.33.3.15 onKeyShortcut . . . 267
7.33.3.16 onKeyUp . . . 267
7.33.3.17 onTouchEvent . . . 267
7.33.3.18 onTrackballEvent . . . 267
7.33.3.19 QtTAG . . . 268
7.33.3.20 stackDeep . . . 268
7.34 org.k4u.customIrcClient.R Class Reference . . . 268
7.34.1 Detailed Description . . . 268
7.35 server::uiClient Class Reference . . . 268
7.35.1 Detailed Description . . . 271
7.35.2 Constructor & Destructor Documentation . . . 271
7.35.2.1 uiClient . . . 271
7.35.3 Member Function Documentation . . . 272
7.35.3.1 changeOwnUser . . . 272
7.35.3.2 chatReceived . . . 272
7.35.3.3 checkForJsonCommand . . . 273
7.35.3.4 disconnected . . . 273
7.35.3.5 getEvents . . . 274
7.35.3.6 receiveMessage . . . 274
7.35.3.7 removeConnection . . . 275

7.35.3.8 send . . . 275
7.35.3.9 userQuery . . . 276
7.35.4 Member Data Documentation . . . 276
7.35.4.1 buffer . . . 276
7.35.4.2 depth . . . 276
7.35.4.3 log . . . 276
7.35.4.4 sock . . . 276
7.36 server::uiServer Class Reference . . . 277
7.36.1 Detailed Description . . . 279
7.36.2 Constructor & Destructor Documentation . . . 279
7.36.2.1 uiServer . . . 279
7.36.2.2 ∼uiServer . . . 280
7.36.3 Member Function Documentation . . . 280
7.36.3.1 acceptConnection . . . 280
7.36.3.2 changeOwnUser . . . 281
7.36.3.3 chatReceived . . . 281
7.36.3.4 doSendEvents . . . 282
7.36.3.5 doUserQuery . . . 282
7.36.3.6 lostConnection . . . 283
7.36.3.7 send . . . 283
7.36.3.8 send . . . 284
7.36.3.9 sgnChangeOwnUserStatus . . . 285
7.36.3.10 sgnChatReceived . . . 285
7.36.3.11 sgnSendEvents . . . 285
7.36.3.12 sgnUserQuery . . . 286
7.36.3.13 startListen . . . 286
7.36.4 Member Data Documentation . . . 286
7.36.4.1 connections . . . 287
7.36.4.2 log . . . 287
7.36.4.3 port . . . 287
7.37 server::ircUser::userColor Struct Reference . . . 287
7.37.1 Detailed Description . . . 287
7.37.2 Member Data Documentation . . . 288
7.37.2.1 b . . . 288
7.37.2.2 g . . . 288
7.37.2.3 r . . . 288

**8 File Documentation** **289**
8.1 C++/customIrcClient/android/gen/org/k4u/customIrcClient/BuildConfig.java File Reference . . . 289
8.2 BuildConfig.java . . . 289

8.3 C++/customIrcClient/android/gen/org/k4u/customIrcClient/R.java File Reference . . . . . 289
8.4 R.java . . . . . 290
8.5 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java File Reference . . . . . 290
8.6 IMinistro.java . . . . . 290
8.7 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java.d File Reference . . . . 292
8.8 IMinistro.java.d . . . . . 292
8.9 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java File Reference 292
8.10 IMinistroCallback.java . . . . . 292
8.11 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java.d File Reference 293
8.12 IMinistroCallback.java.d . . . . . 293
8.13 C++/customIrcClient/android/gen/R.java.d File Reference . . . . . 293
8.14 R.java.d . . . . . 294
8.15 C++/customIrcClient/android/main.cpp File Reference . . . . . 294
8.15.1 Function Documentation . . . . . 295
8.15.1.1 main . . . . . 295
8.16 main.cpp . . . . . 295
8.17 C++/customIrcClient/main.cpp File Reference . . . . . 295
8.17.1 Function Documentation . . . . . 296
8.17.1.1 main . . . . . 296
8.18 main.cpp . . . . . 296
8.19 C++/customIrcServer/main.cpp File Reference . . . . . 296
8.19.1 Function Documentation . . . . . 296
8.19.1.1 main . . . . . 296
8.20 main.cpp . . . . . 297
8.21 C++/customIrcClient/android/mainwindow.cpp File Reference . . . . . 297
8.22 mainwindow.cpp . . . . . 298
8.23 C++/customIrcClient/mainwindow.cpp File Reference . . . . . 299
8.24 mainwindow.cpp . . . . . 299
8.25 C++/customIrcClient/android/mainwindow.h File Reference . . . . . 303
8.26 mainwindow.h . . . . . 303
8.27 C++/customIrcClient/mainwindow.h File Reference . . . . . 304
8.27.1 Macro Definition Documentation . . . . . 305
8.27.1.1 RECONNECT_IN_S . . . . . 305
8.28 mainwindow.h . . . . . 305
8.29 C++/customIrcClient/android/src/org/kde/necessitas/origo/QtActivity.java File Reference . . . . . 306
8.30 QtActivity.java . . . . . 307
8.31 C++/customIrcClient/android/src/org/kde/necessitas/origo/QtApplication.java File Reference . . . . 323
8.32 QtApplication.java . . . . . 323
8.33 C++/customIrcClient/chatbox.cpp File Reference . . . . . 325
8.34 chatbox.cpp . . . . . 326

8.35 C++/customIrcClient/chatbox.h File Reference . . . 327
8.36 chatbox.h . . . 328
8.37 C++/customIrcClient/chatentry.cpp File Reference . . . 329
8.38 chatentry.cpp . . . 329
8.39 C++/customIrcClient/chatentry.h File Reference . . . 331
8.40 chatentry.h . . . 332
8.41 C++/customIrcClient/chatinput.cpp File Reference . . . 332
8.42 chatinput.cpp . . . 333
8.43 C++/customIrcClient/chatinput.h File Reference . . . 333
8.44 chatinput.h . . . 334
8.45 C++/customIrcClient/chatnotification.cpp File Reference . . . 335
8.46 chatnotification.cpp . . . 335
8.47 C++/customIrcClient/chatnotification.h File Reference . . . 336
8.48 chatnotification.h . . . 337
8.49 C++/customIrcClient/clsserverconn.cpp File Reference . . . 338
8.50 clsserverconn.cpp . . . 338
8.51 C++/customIrcClient/clsserverconn.h File Reference . . . 340
8.51.1 Macro Definition Documentation . . . 342
8.51.1.1 MAX_CONNECT_TIMEOUT_IN_MS . . . 342
8.52 clsserverconn.h . . . 342
8.53 C++/customIrcClient/clssettings.cpp File Reference . . . 343
8.54 clssettings.cpp . . . 343
8.55 C++/customIrcServer/clssettings.cpp File Reference . . . 344
8.56 clssettings.cpp . . . 344
8.57 C++/customIrcClient/clssettings.h File Reference . . . 346
8.58 clssettings.h . . . 347
8.59 C++/customIrcServer/clssettings.h File Reference . . . 348
8.59.1 Macro Definition Documentation . . . 349
8.59.1.1 DEFAULT_LISTEN_PORT . . . 349
8.60 clssettings.h . . . 349
8.61 C++/customIrcClient/desktop/clsnotify.cpp File Reference . . . 350
8.62 clsnotify.cpp . . . 350
8.63 C++/customIrcClient/desktop/clsnotify.h File Reference . . . 351
8.64 clsnotify.h . . . 351
8.65 C++/customIrcClient/desktop/clsuserentry.cpp File Reference . . . 352
8.66 clsuserentry.cpp . . . 352
8.67 C++/customIrcClient/desktop/clsuserentry.h File Reference . . . 353
8.68 clsuserentry.h . . . 354
8.69 C++/customIrcClient/desktop/clsuserlist.cpp File Reference . . . 355
8.70 clsuserlist.cpp . . . 355

8.71 C++/customIrcClient/desktop/clsuserlist.h File Reference . . . 356
8.72 clsuserlist.h . . . 357
8.73 C++/customIrcClient/ircuser.cpp File Reference . . . 357
8.74 ircuser.cpp . . . 358
8.75 C++/customIrcServer/ircuser.cpp File Reference . . . 359
8.76 ircuser.cpp . . . 360
8.77 C++/customIrcClient/ircuser.h File Reference . . . 364
8.78 ircuser.h . . . 364
8.79 C++/customIrcServer/ircuser.h File Reference . . . 365
8.80 ircuser.h . . . 366
8.81 C++/customIrcServer/functions.cpp File Reference . . . 368
8.82 functions.cpp . . . 368
8.83 C++/customIrcServer/functions.h File Reference . . . 369
8.84 functions.h . . . 370
8.85 C++/customIrcServer/ircclient.cpp File Reference . . . 371
8.86 ircclient.cpp . . . 371
8.87 C++/customIrcServer/ircclient.h File Reference . . . 375
8.87.1 Macro Definition Documentation . . . 376
8.87.1.1 MAX_CONNECT_TIMEOUT_IN_MS . . . 376
8.87.1.2 PING_CHECK_INTERVAL . . . 376
8.87.1.3 PING_THRESHOLD . . . 377
8.87.1.4 PORT . . . 377
8.88 ircclient.h . . . 377
8.89 C++/customIrcServer/ircCodes.h File Reference . . . 379
8.90 ircCodes.h . . . 383
8.91 C++/customIrcServer/mainapp.cpp File Reference . . . 386
8.92 mainapp.cpp . . . 386
8.93 C++/customIrcServer/mainapp.h File Reference . . . 390
8.93.1 Macro Definition Documentation . . . 391
8.93.1.1 MAX_EVENTS . . . 391
8.93.1.2 RECONNECT_IN_S . . . 391
8.94 mainapp.h . . . 391
8.95 C++/customIrcServer/uiclient.cpp File Reference . . . 392
8.96 uiclient.cpp . . . 392
8.97 C++/customIrcServer/uiclient.h File Reference . . . 393
8.98 uiclient.h . . . 394
8.99 C++/customIrcServer/uiserver.cpp File Reference . . . 395
8.100 uiserver.cpp . . . 396
8.101 C++/customIrcServer/uiserver.h File Reference . . . 397
8.101.1 Macro Definition Documentation . . . 398

8.101.1.1 DEFAULT_LISTENING_PORT . . . 398
8.102uiserver.h . . . 398
8.103C++/resources/clsevent.cpp File Reference . . . 399
8.104clsevent.cpp . . . 399
8.105C++/resources/clsevent.h File Reference . . . 400
8.106clsevent.h . . . 401
8.107C++/resources/clslog.cpp File Reference . . . 402
8.108clslog.cpp . . . 403
8.109C++/resources/clslog.h File Reference . . . 405
8.110clslog.h . . . 406
8.111C++/resources/events/eventchat.cpp File Reference . . . 406
8.112eventchat.cpp . . . 407
8.113C++/resources/events/eventchat.h File Reference . . . 407
8.114eventchat.h . . . 408
8.115C++/resources/events/events.h File Reference . . . 409
8.116events.h . . . 410
8.117C++/resources/events/eventservermessage.cpp File Reference . . . 410
8.118eventservermessage.cpp . . . 411
8.119C++/resources/events/eventservermessage.h File Reference . . . 411
8.120eventservermessage.h . . . 412
8.121C++/resources/events/eventuserchangenick.cpp File Reference . . . 413
8.122eventuserchangenick.cpp . . . 413
8.123C++/resources/events/eventuserchangenick.h File Reference . . . 414
8.124eventuserchangenick.h . . . 415
8.125C++/resources/events/eventuserchangestatus.cpp File Reference . . . 415
8.126eventuserchangestatus.cpp . . . 416
8.127C++/resources/events/eventuserchangestatus.h File Reference . . . 416
8.128eventuserchangestatus.h . . . 417
8.129C++/resources/events/eventuserjoin.cpp File Reference . . . 418
8.130eventuserjoin.cpp . . . 418
8.131C++/resources/events/eventuserjoin.h File Reference . . . 418
8.132eventuserjoin.h . . . 419
8.133C++/resources/events/eventuserleave.cpp File Reference . . . 420
8.134eventuserleave.cpp . . . 420
8.135C++/resources/events/eventuserleave.h File Reference . . . 421
8.136eventuserleave.h . . . 422
8.137C++/resources/json.cpp File Reference . . . 422
8.138json.cpp . . . 423
8.139C++/resources/json.h File Reference . . . 430
8.140json.h . . . 431

8.141 C++/resources/jsoncommand.cpp File Reference . . . . . 432
8.142 jsoncommand.cpp . . . . . 432
8.143 C++/resources/jsoncommand.h File Reference . . . . . 434
8.144 jsoncommand.h . . . . . 435

**Index** **436**

# Chapter 1

# Todo List

**Member server::ircClient::readData ()**

Error handling

**Member server::uiClient::changeOwnUser (jsonCommand &comm)**

Replace the comm object with a QString or NickAndStatus!

**Member server::uiServer::changeOwnUser (jsonCommand &comm)**

Replace jsonCommand by something else

# Chapter 2

# Namespace Index

## 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:


client . . . . . 11
ircCodes . . . . . 11
K4U
Contains functions used for more projects . . . . . 25
org . . . . . 28
org.k4u . . . . . 28
org.k4u.customIrcClient . . . . . 28
org.kde . . . . . 29
org.kde.necessitas . . . . . 29
org.kde.necessitas.ministro . . . . . 29
org.kde.necessitas.origo . . . . . 29
QtJson
A JSON data parser . . . . . 29
server . . . . . 32
shared . . . . . 36
shared::events . . . . . 36
Ui . . . . . 37

# Chapter 3

# Hierarchical Index

## 3.1 Class Hierarchy

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


org.k4u.customIrcClient.BuildConfig . . . 39
shared::events::clsEvent . . . 62
  shared::events::eventChat . . . 116
  shared::events::eventServerMessage . . . 120
  shared::events::eventUserChangeNick . . . 124
  shared::events::eventUserChangeStatus . . . 129
  shared::events::eventUserJoin . . . 134
  shared::events::eventUserLeave . . . 138
K4U::clsLog . . . 69
client::clsSettings . . . 95
server::clsSettings . . . 98
client::chatBox::content . . . 115
IInterface
  org.kde.necessitas.ministro.IMinistro . . . 142
  org.kde.necessitas.ministro.IMinistroCallback . . . 144
server::ircUser . . . 166
server::ircUserList . . . 183
K4U::jsonCommand . . . 189
server::nickAndStatus . . . 230
QFrame
  client::chatEntry . . . 46
QMainWindow
  client::MainWindow . . . 215
  MainWindow . . . 212
QObject
  client::clsNotify . . . 76
  client::clsServerConn . . . 78
  client::ircUser . . . 175
  server::ircClient . . . 146
  server::mainApp . . . 196
  server::uiClient . . . 268
QPlainTextEdit
  client::chatInput . . . 53
QTcpServer
  server::uiServer . . . 277
QWidget
  client::chatBox . . . 40
  client::chatNotification . . . 56

client::clsUserEntry . . . . . 107
client::clsUserList . . . . . 111
org.k4u.customIrcClient.R . . . . . 268
server::ircUser::userColor . . . . . 287
Activity
org.kde.necessitas.origo.QtActivity . . . . . 232
Application
org.kde.necessitas.origo.QtApplication . . . . . 261

# Chapter 4

# Class Index

## 4.1 Class List

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


org.k4u.customIrcClient.BuildConfig . . . 39
client::chatBox
UI Segment containing the chatbox . . . 40
client::chatEntry
Class that holds a chatmessage for UI . . . 46
client::chatInput
UI segment containing the chat input . . . 53
client::chatNotification
UI Segment containing a notification in the chatbox . . . 56
shared::events::clsEvent
The main event class . . . 62
K4U::clsLog
Custom logging class . . . 69
client::clsNotify
Notification class . . . 76
client::clsServerConn
Class that has a connection to the server . . . 78
client::clsSettings
Settings class containing all the variables needed for the client . . . 95
server::clsSettings
Settings class . . . 98
client::clsUserEntry
UI segment for a user entry in the user list . . . 107
client::clsUserList
UI Segment for user list . . . 111
client::chatBox::content . . . 115
shared::events::eventChat
Class containing a chatEvent . . . 116
shared::events::eventServerMessage
Class that contains data for when the server wants to say something . . . 120
shared::events::eventUserChangeNick
Class containing a user who changes his nick-event (nick|status) . . . 124
shared::events::eventUserChangeStatus
Class containing a user who changes his status-event (nick|status) . . . 129
shared::events::eventUserJoin
Class containing a user JOIN event . . . 134
shared::events::eventUserLeave
Class containing a user PART event . . . 138

org.kde.necessitas.ministro.IMinistro . . . 142
org.kde.necessitas.ministro.IMinistroCallback . . . 144
server::ircClient
The ircClient class . . . 146
server::ircUser
The ircUser class . . . 166
client::ircUser
The ircUser class . . . 175
server::ircUserList
Contains all the users . . . 183
K4U::jsonCommand
The jsonCommand class . . . 189
server::mainApp
The main application . . . 196
MainWindow . . . 212
client::MainWindow
Main UI window . . . 215
server::nickAndStatus
The nickAndStatus struct . . . 230
org.kde.necessitas.origo.QtActivity . . . 232
org.kde.necessitas.origo.QtApplication . . . 261
org.k4u.customIrcClient.R . . . 268
server::uiClient
The uiClient class . . . 268
server::uiServer
The uiServer class . . . 277
server::ircUser::userColor
The userColor struct As a replacement for the QColor type . . . 287

# Chapter 5

# File Index

## 5.1 File List

Here is a list of all files with brief descriptions:


C++/customIrcClient/chatbox.cpp . . . 326
C++/customIrcClient/chatbox.h . . . 328
C++/customIrcClient/chatentry.cpp . . . 329
C++/customIrcClient/chatentry.h . . . 332
C++/customIrcClient/chatinput.cpp . . . 333
C++/customIrcClient/chatinput.h . . . 334
C++/customIrcClient/chatnotification.cpp . . . 335
C++/customIrcClient/chatnotification.h . . . 337
C++/customIrcClient/clsserverconn.cpp . . . 338
C++/customIrcClient/clsserverconn.h . . . 342
C++/customIrcClient/clssettings.cpp . . . 343
C++/customIrcClient/clssettings.h . . . 347
C++/customIrcClient/ircuser.cpp . . . 358
C++/customIrcClient/ircuser.h . . . 364
C++/customIrcClient/main.cpp . . . 296
C++/customIrcClient/mainwindow.cpp . . . 299
C++/customIrcClient/mainwindow.h . . . 305
C++/customIrcClient/android/main.cpp . . . 295
C++/customIrcClient/android/mainwindow.cpp . . . 298
C++/customIrcClient/android/mainwindow.h . . . 303
C++/customIrcClient/android/gen/R.java.d . . . 294
C++/customIrcClient/android/gen/org/k4u/customIrcClient/BuildConfig.java . . . 289
C++/customIrcClient/android/gen/org/k4u/customIrcClient/R.java . . . 290
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java . . . 290
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java.d . . . 292
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java . . . 292
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java.d . . . 293
C++/customIrcClient/android/src/org/kde/necessitas/origo/QtActivity.java . . . 307
C++/customIrcClient/android/src/org/kde/necessitas/origo/QtApplication.java . . . 323
C++/customIrcClient/desktop/clsnotify.cpp . . . 350
C++/customIrcClient/desktop/clsnotify.h . . . 351
C++/customIrcClient/desktop/clsuserentry.cpp . . . 352
C++/customIrcClient/desktop/clsuserentry.h . . . 354
C++/customIrcClient/desktop/clsuserlist.cpp . . . 355
C++/customIrcClient/desktop/clsuserlist.h . . . 357
C++/customIrcServer/clssettings.cpp . . . 344
C++/customIrcServer/clssettings.h . . . 349
C++/customIrcServer/functions.cpp . . . 368

C++/customIrcServer/functions.h . . . 370
C++/customIrcServer/ircclient.cpp . . . 371
C++/customIrcServer/ircclient.h . . . 377
C++/customIrcServer/ircCodes.h . . . 383
C++/customIrcServer/ircuser.cpp . . . 360
C++/customIrcServer/ircuser.h . . . 366
C++/customIrcServer/main.cpp . . . 297
C++/customIrcServer/mainapp.cpp . . . 386
C++/customIrcServer/mainapp.h . . . 391
C++/customIrcServer/uiclient.cpp . . . 392
C++/customIrcServer/uiclient.h . . . 394
C++/customIrcServer/uiserver.cpp . . . 396
C++/customIrcServer/uiserver.h . . . 398
C++/resources/clsevent.cpp . . . 399
C++/resources/clsevent.h . . . 401
C++/resources/clslog.cpp . . . 403
C++/resources/clslog.h . . . 406
C++/resources/json.cpp . . . 423
C++/resources/json.h . . . 431
C++/resources/jsoncommand.cpp . . . 432
C++/resources/jsoncommand.h . . . 435
C++/resources/events/eventchat.cpp . . . 407
C++/resources/events/eventchat.h . . . 408
C++/resources/events/events.h . . . 410
C++/resources/events/eventservermessage.cpp . . . 411
C++/resources/events/eventservermessage.h . . . 412
C++/resources/events/eventuserchangenick.cpp . . . 413
C++/resources/events/eventuserchangenick.h . . . 415
C++/resources/events/eventuserchangestatus.cpp . . . 416
C++/resources/events/eventuserchangestatus.h . . . 417
C++/resources/events/eventuserjoin.cpp . . . 418
C++/resources/events/eventuserjoin.h . . . 419
C++/resources/events/eventuserleave.cpp . . . 420
C++/resources/events/eventuserleave.h . . . 422

# Chapter 6

# Namespace Documentation

## 6.1 client Namespace Reference

### Classes

- class chatBox

  *UI Segment containing the chatbox.*
- class chatEntry

  *Class that holds a chatmessage for UI.*
- class chatInput

  *UI segment containing the chat input.*
- class chatNotification

  *UI Segment containing a notification in the chatbox.*
- class clsServerConn

  *Class that has a connection to the server.*
- class clsSettings

  *Settings class containing all the variables needed for the client.*
- class clsNotify

  *Notification class.*
- class clsUserEntry

  *UI segment for a user entry in the user list.*
- class clsUserList

  *UI Segment for user list.*
- class ircUser

  *The ircUser class.*
- class MainWindow

  *Main UI window.*

## 6.2 ircCodes Namespace Reference

### Enumerations

- enum ircCode {
  RPL_WELCOME = 1, RPL_YOURHOST = 2, RPL_CREATED = 3, RPL_MYINFO = 4,
  RPL_ISUPPORT = 5, RPL_SNOMASK = 8, RPL_STATMEMTOT = 9, RPL_BOUNCE = 10,
  RPL_STATMEM = 10, RPL_YOURCOOKIE = 14, RPL_YOURID = 42, RPL_SAVENICK = 43,
  RPL_ATTEMPTINGJUNC = 50, RPL_ATTEMPTINGREROUTE = 51, RPL_TRACELINK = 200, RPL_TRA-

CECONNECTING = 201,
RPL_TRACEHANDSHAKE = 202, RPL_TRACEUNKNOWN = 203, RPL_TRACEOPERATOR = 204, RPL_-TRACEUSER = 205,
RPL_TRACESERVER = 206, RPL_TRACESERVICE = 207, RPL_TRACENEWTYPE = 208, RPL_TRACE-CLASS = 209,
RPL_TRACERECONNECT = 210, RPL_STATS = 210, RPL_STATSLINKINFO = 211, RPL_STATSCOMM-ANDS = 212,
RPL_STATSCLINE = 213, RPL_STATSNLINE = 214, RPL_STATSILINE = 215, RPL_STATSKLINE = 216,
RPL_STATSQLINE = 217, RPL_STATSYLINE = 218, RPL_ENDOFSTATS = 219, RPL_UMODEIS = 221,
RPL_MODLIST = 222, RPL_SQLINE_NICK = 222, RPL_STATSZLINE = 225, RPL_STATSCOUNT = 226,
RPL_SERVICEINFO = 231, RPL_ENDOFSERVICES = 232, RPL_SERVICE = 233, RPL_SERVLIST = 234,
RPL_SERVLISTEND = 235, RPL_STATSVERBOSE = 236, RPL_STATSENGINE = 237, RPL_STATSIAU-TH = 239,
RPL_STATSVLINE = 240, RPL_STATSLLINE = 241, RPL_STATSUPTIME = 242, RPL_STATSOLINE = 243,
RPL_STATSHLINE = 244, RPL_STATSSLINE = 245, RPL_STATSPING = 246, RPL_STATSBLINE = 247,
RPL_STATSDEFINE = 248, RPL_STATSDEBUG = 249, RPL_STATSDLINE = 250, RPL_STATSCONN = 250,
RPL_LUSERCLIENT = 251, RPL_LUSEROP = 252, RPL_LUSERUNKNOWN = 253, RPL_LUSERCHANN-ELS = 254,
RPL_LUSERME = 255, RPL_ADMINME = 256, RPL_ADMINLOC1 = 257, RPL_ADMINLOC2 = 258,
RPL_ADMINEMAIL = 259, RPL_TRACELOG = 261, RPL_TRACEPING = 262, RPL_TRACEEND = 262,
RPL_TRYAGAIN = 263, RPL_LOCALUSERS = 265, RPL_GLOBALUSERS = 266, RPL_START_NETSTAT = 267,
RPL_NETSTAT = 268, RPL_END_NETSTAT = 269, RPL_PRIVS = 270, RPL_SILELIST = 271,
RPL_ENDOFSILELIST = 272, RPL_NOTIFY = 273, RPL_ENDNOTIFY = 274, RPL_STATSDELTA = 274,
RPL_VCHANEXIST = 276, RPL_VCHANLIST = 277, RPL_VCHANHELP = 278, RPL_GLIST = 280,
RPL_ENDOFGLIST = 281, RPL_ACCEPTLIST = 281, RPL_ENDOFACCEPT = 282, RPL_JUPELIST = 282,
RPL_ENDOFJUPELIST = 283, RPL_FEATURE = 284, RPL_GLIST_HASH = 285, RPL_CHANINFO_HAN-DLE = 285,
RPL_NEWHOSTIS = 285, RPL_CHANINFO_USERS = 286, RPL_CHKHEAD = 286, RPL_CHANINFO_CH-OPS = 287,
RPL_CHANUSER = 287, RPL_CHANINFO_VOICES = 288, RPL_PATCHHEAD = 288, RPL_CHANINFO_-AWAY = 289,
RPL_PATCHCON = 289, RPL_CHANINFO_OPERS = 290, RPL_HELPHDR = 290, RPL_DATASTR = 290,
RPL_CHANINFO_BANNED = 291, RPL_HELPOP = 291, RPL_ENDOFCHECK = 291, RPL_CHANINFO_B-ANS = 292,
RPL_HELPTLR = 292, RPL_CHANINFO_INVITE = 293, RPL_HELPHLP = 293, RPL_CHANINFO_INVITES = 294,
RPL_HELPFWD = 294, RPL_CHANINFO_KICK = 295, RPL_HELPIGN = 295, RPL_CHANINFO_KICKS = 296,
RPL_END_CHANINFO = 299, RPL_NONE = 300, RPL_AWAY = 301, RPL_USERHOST = 302,
RPL_ISON = 303, RPL_TEXT = 304, RPL_UNAWAY = 305, RPL_NOWAWAY = 306,
RPL_WHOISREGNICK = 307, RPL_SUSERHOST = 307, RPL_NOTIFYACTION = 308, RPL_WHOISADM-IN = 308,
RPL_NICKTRACE = 309, RPL_WHOISSADMIN = 309, RPL_WHOISHELPER = 309, RPL_WHOISSVCMS-G = 310,
RPL_WHOISHELPOP = 310, RPL_WHOISSERVICE = 310, RPL_WHOISUSER = 311, RPL_WHOISSERV-ER = 312,
RPL_WHOISOPERATOR = 313, RPL_WHOWASUSER = 314, RPL_ENDOFWHO = 315, RPL_WHOISCH-ANOP = 316,
RPL_WHOISIDLE = 317, RPL_ENDOFWHOIS = 318, RPL_WHOISCHANNELS = 319, RPL_WHOISVIRT = 320,
RPL_WHOIS_HIDDEN = 320, RPL_WHOISSPECIAL = 320, RPL_LISTSTART = 321, RPL_LIST = 322,
RPL_LISTEND = 323, RPL_CHANNELMODEIS = 324, RPL_UNIQOPIS = 325, RPL_CHANNELPASSIS = 325,
RPL_NOCHANPASS = 326, RPL_CHPASSUNKNOWN = 327, RPL_CHANNEL_URL = 328, RPL_CREATI-

ONTIME = 329,
RPL_WHOWAS_TIME = 330, RPL_WHOISACCOUNT = 330, RPL_NOTOPIC = 331, RPL_TOPIC = 332, RPL_TOPICWHOTIME = 333, RPL_LISTUSAGE = 334, RPL_COMMANDSYNTAX = 334, RPL_LISTSYNTAX = 334,
RPL_CHANPASSOK = 338, RPL_WHOISACTUALLY = 338, RPL_BADCHANPASS = 339, RPL_INVITING = 341,
RPL_SUMMONING = 342, RPL_INVITED = 345, RPL_INVITELIST = 346, RPL_ENDOFINVITELIST = 347, RPL_EXCEPTLIST = 348, RPL_ENDOFEXCEPTLIST = 349, RPL_VERSION = 351, RPL_WHOREPLY = 352,
RPL_NAMREPLY = 353, RPL_WHOSPCRPL = 354, RPL_NAMREPLY_ = 355, RPL_KILLDONE = 361, RPL_CLOSING = 362, RPL_CLOSEEND = 363, RPL_LINKS = 364, RPL_ENDOFLINKS = 365, RPL_ENDOFNAMES = 366, RPL_BANLIST = 367, RPL_ENDOFBANLIST = 368, RPL_ENDOFWHOWAS = 369,
RPL_INFO = 371, RPL_MOTD = 372, RPL_INFOSTART = 373, RPL_ENDOFINFO = 374, RPL_MOTDSTART = 375, RPL_ENDOFMOTD = 376, RPL_KICKEXPIRED = 377, RPL_SPAM = 377, RPL_BANEXPIRED = 378, RPL_WHOISHOST = 378, RPL_KICKLINKED = 379, RPL_WHOISMODES = 379,
RPL_BANLINKED = 380, RPL_YOURHELPER = 380, RPL_YOUREOPER = 381, RPL_REHASHING = 382, RPL_YOURESERVICE = 383, RPL_MYPORTIS = 384, RPL_NOTOPERANYMORE = 385, RPL_QLIST = 386,
RPL_IRCOPS = 386, RPL_ENDOFQLIST = 387, RPL_ENDOFIRCOPS = 387, RPL_ALIST = 388, RPL_ENDOFALIST = 389, RPL_TIME = 391, RPL_USERSSTART = 392, RPL_USERS = 393, RPL_ENDOFUSERS = 394, RPL_NOUSERS = 395, RPL_HOSTHIDDEN = 396, ERR_UNKNOWNERROR = 400,
ERR_NOSUCHNICK = 401, ERR_NOSUCHSERVER = 402, ERR_NOSUCHCHANNEL = 403, ERR_CANNOTSENDTOCHAN = 404,
ERR_TOOMANYCHANNELS = 405, ERR_WASNOSUCHNICK = 406, ERR_TOOMANYTARGETS = 407, ERR_NOSUCHSERVICE = 408,
ERR_NOCOLORSONCHAN = 408, ERR_NOORIGIN = 409, ERR_NORECIPIENT = 411, ERR_NOTEXTTOSEND = 412,
ERR_NOTOPLEVEL = 413, ERR_WILDTOPLEVEL = 414, ERR_BADMASK = 415, ERR_TOOMANYMATCHES = 416,
ERR_QUERYTOOLONG = 416, ERR_LENGTHTRUNCATED = 419, ERR_UNKNOWNCOMMAND = 421, ERR_NOMOTD = 422,
ERR_NOADMININFO = 423, ERR_FILEERROR = 424, ERR_NOOPERMOTD = 425, ERR_TOOMANYAWAY = 429,
ERR_EVENTNICKCHANGE = 430, ERR_NONICKNAMEGIVEN = 431, ERR_ERRONEUSNICKNAME = 432, ERR_NICKNAMEINUSE = 433,
ERR_SERVICENAMEINUSE = 434, ERR_NORULES = 434, ERR_SERVICECONFUSED = 435, ERR_BANONCHAN = 435,
ERR_NICKCOLLISION = 436, ERR_UNAVAILRESOURCE = 437, ERR_BANNICKCHANGE = 437, ERR_NICKTOOFAST = 438,
ERR_DEAD = 438, ERR_TARGETTOOFAST = 439, ERR_SERVICESDOWN = 440, ERR_USERNOTINCHANNEL = 441,
ERR_NOTONCHANNEL = 442, ERR_USERONCHANNEL = 443, ERR_NOLOGIN = 444, ERR_SUMMONDISABLED = 445,
ERR_USERSDISABLED = 446, ERR_NONICKCHANGE = 447, ERR_NOTIMPLEMENTED = 449, ERR_NOTREGISTERED = 451,
ERR_IDCOLLISION = 452, ERR_NICKLOST = 453, ERR_HOSTILENAME = 455, ERR_ACCEPTFULL = 456,
ERR_ACCEPTEXIST = 457, ERR_ACCEPTNOT = 458, ERR_NOHIDING = 459, ERR_NOTFORHALFOPS = 460,
ERR_NEEDMOREPARAMS = 461, ERR_ALREADYREGISTERED = 462, ERR_NOPERMFORHOST = 463, ERR_PASSWDMISMATCH = 464,
ERR_YOUREBANNEDCREEP = 465, ERR_YOUWILLBEBANNED = 466, ERR_KEYSET = 467, ERR_INVALIDUSERNAME = 468,
ERR_ONLYSERVERSCANCHANGE = 468, ERR_LINKSET = 469, ERR_LINKCHANNEL = 470, ERR_KIC-

KEDFROMCHAN = 470,
ERR_CHANNELISFULL = 471, ERR_UNKNOWNMODE = 472, ERR_INVITEONLYCHAN = 473, ERR_BANNEDFROMCHAN = 474,
ERR_BADCHANNELKEY = 475, ERR_BADCHANMASK = 476, ERR_NOCHANMODES = 477, ERR_NEEDREGGEDNICK = 477,
ERR_BANLISTFULL = 478, ERR_BADCHANNAME = 479, ERR_LINKFAIL = 479, ERR_NOULINE = 480,
ERR_CANNOTKNOCK = 480, ERR_NOPRIVILEGES = 481, ERR_CHANOPRIVSNEEDED = 482, ERR_CANTKILLSERVER = 483,
ERR_RESTRICTED = 484, ERR_ISCHANSERVICE = 484, ERR_DESYNC = 484, ERR_ATTACKDENY = 484,
ERR_UNIQOPRIVSNEEDED = 485, ERR_KILLDENY = 485, ERR_CANTKICKADMIN = 485, ERR_ISREALSERVICE = 485,
ERR_NONONREG = 486, ERR_HTMDISABLED = 486, ERR_ACCOUNTONLY = 486, ERR_CHANTOORECENT = 487,
ERR_MSGSERVICES = 487, ERR_TSLESSCHAN = 488, ERR_VOICENEEDED = 489, ERR_SECUREONLYCHAN = 489,
ERR_NOOPERHOST = 491, ERR_NOSERVICEHOST = 492, ERR_NOFEATURE = 493, ERR_BADFEATURE = 494,
ERR_BADLOGTYPE = 495, ERR_BADLOGSYS = 496, ERR_BADLOGVALUE = 497, ERR_ISOPERLCHAN = 498,
ERR_CHANOWNPRIVNEEDED = 499, ERR_UMODEUNKNOWNFLAG = 501, ERR_USERSDONTMATCH = 502, ERR_GHOSTEDCLIENT = 503,
ERR_VWORLDWARN = 503, ERR_USERNOTONSERV = 504, ERR_SILELISTFULL = 511, ERR_TOOMANYWATCH = 512,
ERR_BADPING = 513, ERR_INVALID_ERROR = 514, ERR_TOOMANYDCC = 514, ERR_BADEXPIRE = 515,
ERR_DONTCHEAT = 516, ERR_DISABLED = 517, ERR_NOINVITE = 518, ERR_LONGMASK = 518,
ERR_ADMONLY = 519, ERR_TOOMANYUSERS = 519, ERR_OPERONLY = 520, ERR_MASKTOOWIDE = 520,
ERR_WHOTRUNC = 520, ERR_LISTSYNTAX = 521, ERR_WHOSYNTAX = 522, ERR_WHOLIMEXCEED = 523,
ERR_QUARANTINED = 524, ERR_OPERSPVERIFY = 524, ERR_REMOTEPFX = 525, ERR_PFXUNROUTABLE = 526,
ERR_BADHOSTMASK = 550, ERR_HOSTUNAVAIL = 551, ERR_USINGSLINE = 552, ERR_STATSSLINE = 553,
RPL_LOGON = 600, RPL_LOGOFF = 601, RPL_WATCHOFF = 602, RPL_WATCHSTAT = 603,
RPL_NOWON = 604, RPL_NOWOFF = 605, RPL_WATCHLIST = 606, RPL_ENDOFWATCHLIST = 607,
RPL_WATCHCLEAR = 608, RPL_ISOPER = 610, RPL_ISLOCOP = 611, RPL_ISNOTOPER = 612,
RPL_ENDOFISOPER = 613, RPL_DCCSTATUS = 617, RPL_DCCLIST = 618, RPL_ENDOFDCCLIST = 619,
RPL_WHOWASHOST = 619, RPL_DCCINFO = 620, RPL_ENDOFO = 626, RPL_SETTINGS = 630,
RPL_ENDOFSETTINGS = 631, RPL_DUMPING = 640, RPL_DUMPRPL = 641, RPL_EODUMP = 642,
RPL_TRACEROUTE_HOP = 660, RPL_TRACEROUTE_START = 661, RPL_MODECHANGEWARN = 662, RPL_CHANREDIR = 663,
RPL_SERVMODEIS = 664, RPL_OTHERUMODEIS = 665, RPL_ENDOF_GENERIC = 666, RPL_WHOWASDETAILS = 670,
RPL_WHOISSECURE = 671, RPL_UNKNOWNMODES = 672, RPL_CANNOTSETMODES = 673, RPL_LUSERSTAFF = 678,
RPL_TIMEONSERVERIS = 679, RPL_NETWORKS = 682, RPL_YOURLANGUAGEIS = 687, RPL_LANGUAGE = 688,
RPL_WHOISSTAFF = 689, RPL_WHOISLANGUAGE = 690, RPL_HELPSTART = 704, RPL_HELPTXT = 705,
RPL_ENDOFHELP = 706, RPL_ETRACEFULL = 708, RPL_ETRACE = 709, RPL_KNOCK = 710,
RPL_KNOCKDLVR = 711, ERR_TOOMANYKNOCK = 712, ERR_CHANOPEN = 713, ERR_KNOCKONCHAN = 714,
ERR_KNOCKDISABLED = 715, RPL_TARGUMODEG = 716, RPL_TARGNOTIFY = 717, RPL_UMODEG-

MSG = 718,
RPL_ENDOFOMOTD = 722, ERR_NOPRIVS = 723, RPL_TESTMARK = 724, RPL_TESTLINE = 725, RPL_NOTESTLINE = 726, RPL_XINFO = 771, RPL_XINFOSTART = 773, RPL_XINFOEND = 774, ERR_CANNOTDOCOMMAND = 972, ERR_CANNOTCHANGEUMODE = 973, ERR_CANNOTCHANGECHANMODE = 974, ERR_CANNOTCHANGESERVERMODE = 975,
ERR_CANNOTSENDTONICK = 976, ERR_UNKNOWNSERVERMODE = 977, ERR_SERVERMODELOCK = 979, ERR_BADCHARENCODING = 980,
ERR_TOOMANYLANGUAGES = 981, ERR_NOLANGUAGE = 982, ERR_TEXTTOOSHORT = 983, ERR_NUMERIC_ERR = 999 }

### 6.2.1 Enumeration Type Documentation

#### 6.2.1.1 enum ircCodes::ircCode

**Enumerator**

***RPL_WELCOME***

***RPL_YOURHOST***

***RPL_CREATED***

***RPL_MYINFO***

***RPL_ISUPPORT***

***RPL_SNOMASK***

***RPL_STATMEMTOT***

***RPL_BOUNCE***

***RPL_STATMEM***

***RPL_YOURCOOKIE***

***RPL_YOURID***

***RPL_SAVENICK***

***RPL_ATTEMPTINGJUNC***

***RPL_ATTEMPTINGREROUTE***

***RPL_TRACELINK***

***RPL_TRACECONNECTING***

***RPL_TRACEHANDSHAKE***

***RPL_TRACEUNKNOWN***

***RPL_TRACEOPERATOR***

***RPL_TRACEUSER***

***RPL_TRACESERVER***

***RPL_TRACESERVICE***

***RPL_TRACENEWTYPE***

***RPL_TRACECLASS***

***RPL_TRACERECONNECT***

***RPL_STATS***

***RPL_STATSLINKINFO***

***RPL_STATSCOMMANDS***

***RPL_STATSCLINE***

***RPL_STATSNLINE***

***RPL_STATSILINE***

***RPL_STATSKLINE***

*RPL_STATSQLINE*
*RPL_STATSYLINE*
*RPL_ENDOFSTATS*
*RPL_UMODEIS*
*RPL_MODLIST*
*RPL_SQLINE_NICK*
*RPL_STATSZLINE*
*RPL_STATSCOUNT*
*RPL_SERVICEINFO*
*RPL_ENDOFSERVICES*
*RPL_SERVICE*
*RPL_SERVLIST*
*RPL_SERVLISTEND*
*RPL_STATSVERBOSE*
*RPL_STATSENGINE*
*RPL_STATSIAUTH*
*RPL_STATSVLINE*
*RPL_STATSLLINE*
*RPL_STATSUPTIME*
*RPL_STATSOLINE*
*RPL_STATSHLINE*
*RPL_STATSSLINE*
*RPL_STATSPING*
*RPL_STATSBLINE*
*RPL_STATSDEFINE*
*RPL_STATSDEBUG*
*RPL_STATSDLINE*
*RPL_STATSCONN*
*RPL_LUSERCLIENT*
*RPL_LUSEROP*
*RPL_LUSERUNKNOWN*
*RPL_LUSERCHANNELS*
*RPL_LUSERME*
*RPL_ADMINME*
*RPL_ADMINLOC1*
*RPL_ADMINLOC2*
*RPL_ADMINEMAIL*
*RPL_TRACELOG*
*RPL_TRACEPING*
*RPL_TRACEEND*
*RPL_TRYAGAIN*
*RPL_LOCALUSERS*
*RPL_GLOBALUSERS*
*RPL_START_NETSTAT*
*RPL_NETSTAT*

*RPL_END_NETSTAT*
*RPL_PRIVS*
*RPL_SILELIST*
*RPL_ENDOFSILELIST*
*RPL_NOTIFY*
*RPL_ENDNOTIFY*
*RPL_STATSDELTA*
*RPL_VCHANEXIST*
*RPL_VCHANLIST*
*RPL_VCHANHELP*
*RPL_GLIST*
*RPL_ENDOFGLIST*
*RPL_ACCEPTLIST*
*RPL_ENDOFACCEPT*
*RPL_JUPELIST*
*RPL_ENDOFJUPELIST*
*RPL_FEATURE*
*RPL_GLIST_HASH*
*RPL_CHANINFO_HANDLE*
*RPL_NEWHOSTIS*
*RPL_CHANINFO_USERS*
*RPL_CHKHEAD*
*RPL_CHANINFO_CHOPS*
*RPL_CHANUSER*
*RPL_CHANINFO_VOICES*
*RPL_PATCHHEAD*
*RPL_CHANINFO_AWAY*
*RPL_PATCHCON*
*RPL_CHANINFO_OPERS*
*RPL_HELPHDR*
*RPL_DATASTR*
*RPL_CHANINFO_BANNED*
*RPL_HELPOP*
*RPL_ENDOFCHECK*
*RPL_CHANINFO_BANS*
*RPL_HELPTLR*
*RPL_CHANINFO_INVITE*
*RPL_HELPHLP*
*RPL_CHANINFO_INVITES*
*RPL_HELPFWD*
*RPL_CHANINFO_KICK*
*RPL_HELPIGN*
*RPL_CHANINFO_KICKS*
*RPL_END_CHANINFO*
*RPL_NONE*

*RPL_AWAY*
*RPL_USERHOST*
*RPL_ISON*
*RPL_TEXT*
*RPL_UNAWAY*
*RPL_NOWAWAY*
*RPL_WHOISREGNICK*
*RPL_SUSERHOST*
*RPL_NOTIFYACTION*
*RPL_WHOISADMIN*
*RPL_NICKTRACE*
*RPL_WHOISSADMIN*
*RPL_WHOISHELPER*
*RPL_WHOISSVCMSG*
*RPL_WHOISHELPOP*
*RPL_WHOISSERVICE*
*RPL_WHOISUSER*
*RPL_WHOISSERVER*
*RPL_WHOISOPERATOR*
*RPL_WHOWASUSER*
*RPL_ENDOFWHO*
*RPL_WHOISCHANOP*
*RPL_WHOISIDLE*
*RPL_ENDOFWHOIS*
*RPL_WHOISCHANNELS*
*RPL_WHOISVIRT*
*RPL_WHOIS_HIDDEN*
*RPL_WHOISSPECIAL*
*RPL_LISTSTART*
*RPL_LIST*
*RPL_LISTEND*
*RPL_CHANNELMODEIS*
*RPL_UNIQOPIS*
*RPL_CHANNELPASSIS*
*RPL_NOCHANPASS*
*RPL_CHPASSUNKNOWN*
*RPL_CHANNEL_URL*
*RPL_CREATIONTIME*
*RPL_WHOWAS_TIME*
*RPL_WHOISACCOUNT*
*RPL_NOTOPIC*
*RPL_TOPIC*
*RPL_TOPICWHOTIME*
*RPL_LISTUSAGE*
*RPL_COMMANDSYNTAX*

*RPL_LISTSYNTAX*
*RPL_CHANPASSOK*
*RPL_WHOISACTUALLY*
*RPL_BADCHANPASS*
*RPL_INVITING*
*RPL_SUMMONING*
*RPL_INVITED*
*RPL_INVITELIST*
*RPL_ENDOFINVITELIST*
*RPL_EXCEPTLIST*
*RPL_ENDOFEXCEPTLIST*
*RPL_VERSION*
*RPL_WHOREPLY*
*RPL_NAMREPLY*
*RPL_WHOSPCRPL*
*RPL_NAMREPLY_*
*RPL_KILLDONE*
*RPL_CLOSING*
*RPL_CLOSEEND*
*RPL_LINKS*
*RPL_ENDOFLINKS*
*RPL_ENDOFNAMES*
*RPL_BANLIST*
*RPL_ENDOFBANLIST*
*RPL_ENDOFWHOWAS*
*RPL_INFO*
*RPL_MOTD*
*RPL_INFOSTART*
*RPL_ENDOFINFO*
*RPL_MOTDSTART*
*RPL_ENDOFMOTD*
*RPL_KICKEXPIRED*
*RPL_SPAM*
*RPL_BANEXPIRED*
*RPL_WHOISHOST*
*RPL_KICKLINKED*
*RPL_WHOISMODES*
*RPL_BANLINKED*
*RPL_YOURHELPER*
*RPL_YOUREOPER*
*RPL_REHASHING*
*RPL_YOURESERVICE*
*RPL_MYPORTIS*
*RPL_NOTOPERANYMORE*
*RPL_QLIST*

*RPL_IRCOPS*
*RPL_ENDOFQLIST*
*RPL_ENDOFIRCOPS*
*RPL_ALIST*
*RPL_ENDOFALIST*
*RPL_TIME*
*RPL_USERSSTART*
*RPL_USERS*
*RPL_ENDOFUSERS*
*RPL_NOUSERS*
*RPL_HOSTHIDDEN*
*ERR_UNKNOWNERROR*
*ERR_NOSUCHNICK*
*ERR_NOSUCHSERVER*
*ERR_NOSUCHCHANNEL*
*ERR_CANNOTSENDTOCHAN*
*ERR_TOOMANYCHANNELS*
*ERR_WASNOSUCHNICK*
*ERR_TOOMANYTARGETS*
*ERR_NOSUCHSERVICE*
*ERR_NOCOLORSONCHAN*
*ERR_NOORIGIN*
*ERR_NORECIPIENT*
*ERR_NOTEXTTOSEND*
*ERR_NOTOPLEVEL*
*ERR_WILDTOPLEVEL*
*ERR_BADMASK*
*ERR_TOOMANYMATCHES*
*ERR_QUERYTOOLONG*
*ERR_LENGTHTRUNCATED*
*ERR_UNKNOWNCOMMAND*
*ERR_NOMOTD*
*ERR_NOADMININFO*
*ERR_FILEERROR*
*ERR_NOOPERMOTD*
*ERR_TOOMANYAWAY*
*ERR_EVENTNICKCHANGE*
*ERR_NONICKNAMEGIVEN*
*ERR_ERRONEUSNICKNAME*
*ERR_NICKNAMEINUSE*
*ERR_SERVICENAMEINUSE*
*ERR_NORULES*
*ERR_SERVICECONFUSED*
*ERR_BANONCHAN*
*ERR_NICKCOLLISION*

*ERR_UNAVAILRESOURCE*
*ERR_BANNICKCHANGE*
*ERR_NICKTOOFAST*
*ERR_DEAD*
*ERR_TARGETTOOFAST*
*ERR_SERVICESDOWN*
*ERR_USERNOTINCHANNEL*
*ERR_NOTONCHANNEL*
*ERR_USERONCHANNEL*
*ERR_NOLOGIN*
*ERR_SUMMONDISABLED*
*ERR_USERSDISABLED*
*ERR_NONICKCHANGE*
*ERR_NOTIMPLEMENTED*
*ERR_NOTREGISTERED*
*ERR_IDCOLLISION*
*ERR_NICKLOST*
*ERR_HOSTILENAME*
*ERR_ACCEPTFULL*
*ERR_ACCEPTEXIST*
*ERR_ACCEPTNOT*
*ERR_NOHIDING*
*ERR_NOTFORHALFOPS*
*ERR_NEEDMOREPARAMS*
*ERR_ALREADYREGISTERED*
*ERR_NOPERMFORHOST*
*ERR_PASSWDMISMATCH*
*ERR_YOUREBANNEDCREEP*
*ERR_YOUWILLBEBANNED*
*ERR_KEYSET*
*ERR_INVALIDUSERNAME*
*ERR_ONLYSERVERSCANCHANGE*
*ERR_LINKSET*
*ERR_LINKCHANNEL*
*ERR_KICKEDFROMCHAN*
*ERR_CHANNELISFULL*
*ERR_UNKNOWNMODE*
*ERR_INVITEONLYCHAN*
*ERR_BANNEDFROMCHAN*
*ERR_BADCHANNELKEY*
*ERR_BADCHANMASK*
*ERR_NOCHANMODES*
*ERR_NEEDREGGEDNICK*
*ERR_BANLISTFULL*
*ERR_BADCHANNAME*

*ERR_LINKFAIL*
*ERR_NOULINE*
*ERR_CANNOTKNOCK*
*ERR_NOPRIVILEGES*
*ERR_CHANOPRIVSNEEDED*
*ERR_CANTKILLSERVER*
*ERR_RESTRICTED*
*ERR_ISCHANSERVICE*
*ERR_DESYNC*
*ERR_ATTACKDENY*
*ERR_UNIQOPRIVSNEEDED*
*ERR_KILLDENY*
*ERR_CANTKICKADMIN*
*ERR_ISREALSERVICE*
*ERR_NONONREG*
*ERR_HTMDISABLED*
*ERR_ACCOUNTONLY*
*ERR_CHANTOORECENT*
*ERR_MSGSERVICES*
*ERR_TSLESSCHAN*
*ERR_VOICENEEDED*
*ERR_SECUREONLYCHAN*
*ERR_NOOPERHOST*
*ERR_NOSERVICEHOST*
*ERR_NOFEATURE*
*ERR_BADFEATURE*
*ERR_BADLOGTYPE*
*ERR_BADLOGSYS*
*ERR_BADLOGVALUE*
*ERR_ISOPERLCHAN*
*ERR_CHANOWNPRIVNEEDED*
*ERR_UMODEUNKNOWNFLAG*
*ERR_USERSDONTMATCH*
*ERR_GHOSTEDCLIENT*
*ERR_VWORLDWARN*
*ERR_USERNOTONSERV*
*ERR_SILELISTFULL*
*ERR_TOOMANYWATCH*
*ERR_BADPING*
*ERR_INVALID_ERROR*
*ERR_TOOMANYDCC*
*ERR_BADEXPIRE*
*ERR_DONTCHEAT*
*ERR_DISABLED*
*ERR_NOINVITE*

*ERR_LONGMASK*
*ERR_ADMONLY*
*ERR_TOOMANYUSERS*
*ERR_OPERONLY*
*ERR_MASKTOOWIDE*
*ERR_WHOTRUNC*
*ERR_LISTSYNTAX*
*ERR_WHOSYNTAX*
*ERR_WHOLIMEXCEED*
*ERR_QUARANTINED*
*ERR_OPERSPVERIFY*
*ERR_REMOTEPFX*
*ERR_PFXUNROUTABLE*
*ERR_BADHOSTMASK*
*ERR_HOSTUNAVAIL*
*ERR_USINGSLINE*
*ERR_STATSSLINE*
*RPL_LOGON*
*RPL_LOGOFF*
*RPL_WATCHOFF*
*RPL_WATCHSTAT*
*RPL_NOWON*
*RPL_NOWOFF*
*RPL_WATCHLIST*
*RPL_ENDOFWATCHLIST*
*RPL_WATCHCLEAR*
*RPL_ISOPER*
*RPL_ISLOCOP*
*RPL_ISNOTOPER*
*RPL_ENDOFISOPER*
*RPL_DCCSTATUS*
*RPL_DCCLIST*
*RPL_ENDOFDCCLIST*
*RPL_WHOWASHOST*
*RPL_DCCINFO*
*RPL_ENDOFO*
*RPL_SETTINGS*
*RPL_ENDOFSETTINGS*
*RPL_DUMPING*
*RPL_DUMPRPL*
*RPL_EODUMP*
*RPL_TRACEROUTE_HOP*
*RPL_TRACEROUTE_START*
*RPL_MODECHANGEWARN*
*RPL_CHANREDIR*

***RPL_SERVMODEIS***
***RPL_OTHERUMODEIS***
***RPL_ENDOF_GENERIC***
***RPL_WHOWASDETAILS***
***RPL_WHOISSECURE***
***RPL_UNKNOWNMODES***
***RPL_CANNOTSETMODES***
***RPL_LUSERSTAFF***
***RPL_TIMEONSERVERIS***
***RPL_NETWORKS***
***RPL_YOURLANGUAGEIS***
***RPL_LANGUAGE***
***RPL_WHOISSTAFF***
***RPL_WHOISLANGUAGE***
***RPL_HELPSTART***
***RPL_HELPTXT***
***RPL_ENDOFHELP***
***RPL_ETRACEFULL***
***RPL_ETRACE***
***RPL_KNOCK***
***RPL_KNOCKDLVR***
***ERR_TOOMANYKNOCK***
***ERR_CHANOPEN***
***ERR_KNOCKONCHAN***
***ERR_KNOCKDISABLED***
***RPL_TARGUMODEG***
***RPL_TARGNOTIFY***
***RPL_UMODEGMSG***
***RPL_ENDOFOMOTD***
***ERR_NOPRIVS***
***RPL_TESTMARK***
***RPL_TESTLINE***
***RPL_NOTESTLINE***
***RPL_XINFO***
***RPL_XINFOSTART***
***RPL_XINFOEND***
***ERR_CANNOTDOCOMMAND***
***ERR_CANNOTCHANGEUMODE***
***ERR_CANNOTCHANGECHANMODE***
***ERR_CANNOTCHANGESERVERMODE***
***ERR_CANNOTSENDTONICK***
***ERR_UNKNOWNSERVERMODE***
***ERR_SERVERMODELOCK***
***ERR_BADCHARENCODING***
***ERR_TOOMANYLANGUAGES***
***ERR_NOLANGUAGE***
***ERR_TEXTTOOSHORT***
***ERR_NUMERIC_ERR***

Definition at line 5 of file ircCodes.h.

## 6.3 K4U Namespace Reference

Contains functions used for more projects.

### Classes

- class clsLog

  *Custom logging class.*
- class jsonCommand

  *The jsonCommand class.*

### Enumerations

- enum logTags {
  LOGTAGS_UNKNOWN = 0xFF, LOGTAGS_IRC = 0x00, LOGTAGS_MAIN, LOGTAGS_USER, LOGTAGS_SERVER, LOGTAGS_UI, LOGTAGS_WEB, LOGTAGS_EVENT }
- enum jsonCommands {
  JSONCOMMAND_NONE = 0x00, JSONCOMMAND_PING = 0x01, JSONCOMMAND_EVENT, JSONCOMMAND_USERINFOREQ,
  JSONCOMMAND_USERQUERY, JSONCOMMAND_OWNCHAT, JSONCOMMAND_CHANGEOWNUSER, JSONCOMMAND_GETEVENTS }

### Functions

- clsLog & endl (clsLog &log)
- clsLog & operator<< (clsLog &log, jsonCommand &command)
- clsLog & operator<< (clsLog &log, const jsonCommand &command)

### 6.3.1 Detailed Description

Contains functions used for more projects.

**Author**

Koen Beckers (K-4U)

### 6.3.2 Enumeration Type Documentation

#### 6.3.2.1 enum K4U::jsonCommands

**Enumerator**

***JSONCOMMAND_NONE***

***JSONCOMMAND_PING***

***JSONCOMMAND_EVENT***

***JSONCOMMAND_USERINFOREQ***

***JSONCOMMAND_USERQUERY***

***JSONCOMMAND_OWNCHAT***

***JSONCOMMAND_CHANGEOWNUSER***

***JSONCOMMAND_GETEVENTS***

Definition at line 15 of file jsoncommand.h.

#### 6.3.2.2 enum K4U::logTags

**Enumerator**

***LOGTAGS_UNKNOWN***

***LOGTAGS_IRC***

***LOGTAGS_MAIN***

***LOGTAGS_USER***

***LOGTAGS_SERVER***

***LOGTAGS_UI***

***LOGTAGS_WEB***

***LOGTAGS_EVENT***

Definition at line 10 of file clslog.h.

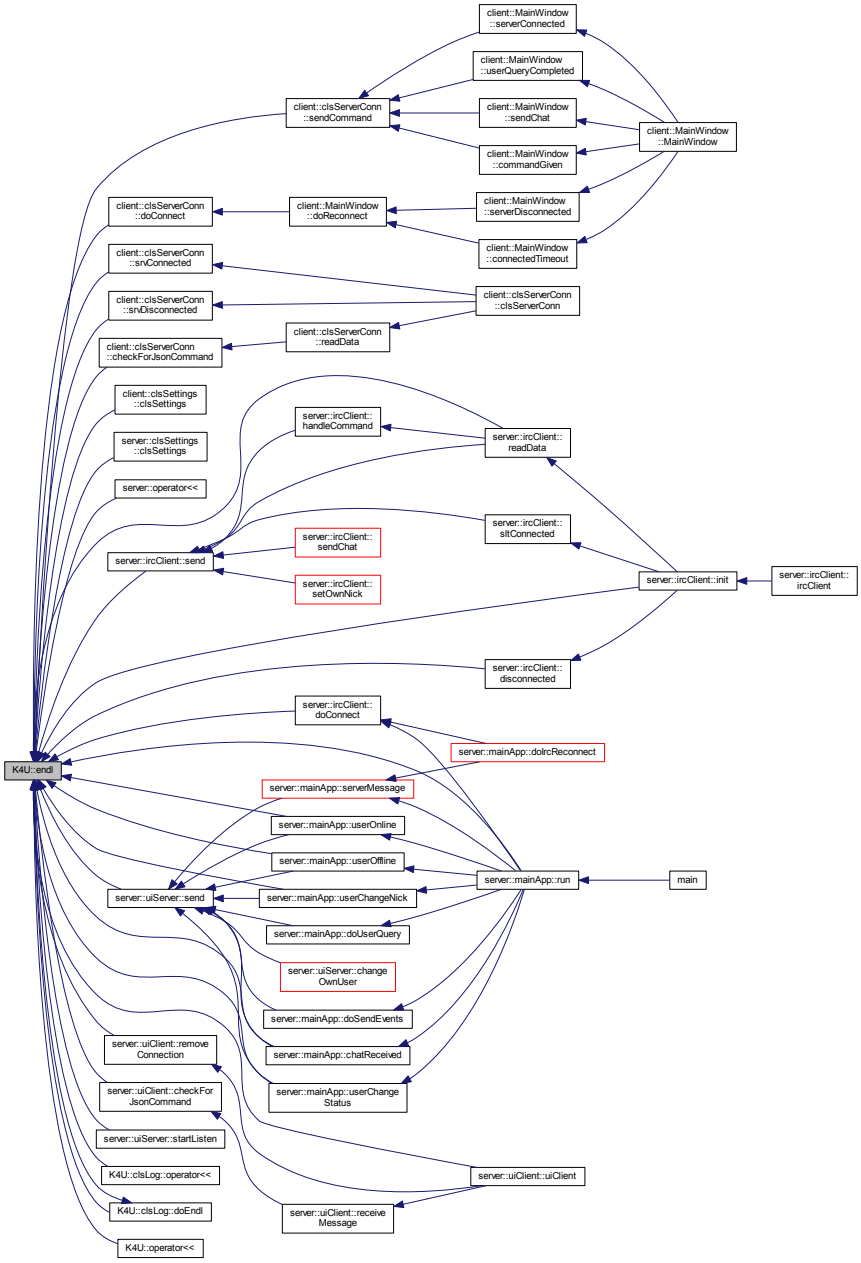
### 6.3.3 Function Documentation

#### 6.3.3.1 clsLog & K4U::endl ( clsLog & *log* )

Definition at line 65 of file clslog.cpp.

Here is the call graph for this function:

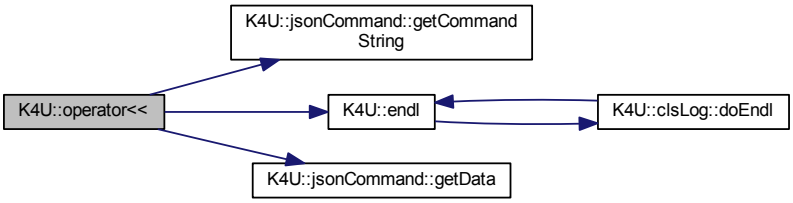
Here is the caller graph for this function:




#### 6.3.3.2 clsLog& K4U::operator<< ( clsLog & *log*, jsonCommand & *command* )

Definition at line 110 of file jsoncommand.cpp.

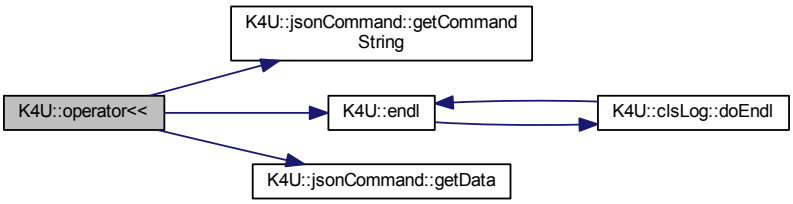
Here is the call graph for this function:




#### 6.3.3.3 clsLog& K4U::operator<< ( clsLog & *log*, const jsonCommand & *command* )

Definition at line 117 of file jsoncommand.cpp.

Here is the call graph for this function:




## 6.4 Package org

### Packages

- package k4u
- package kde

## 6.5 Package org.k4u

### Packages

- package customIrcClient

## 6.6 Package org.k4u.customIrcClient

### Classes

- class BuildConfig
- class R

### 6.6.1 Detailed Description

Automatically generated file. DO NOT MODIFY

## 6.7 Package org.kde

### Packages

- package necessitas

## 6.8 Package org.kde.necessitas

### Packages

- package ministro
- package origo

## 6.9 Package org.kde.necessitas.ministro

### Classes

- interface IMinistro
- interface IMinistroCallback

## 6.10 Package org.kde.necessitas.origo

### Classes

- class QtActivity
- class QtApplication

## 6.11 QtJson Namespace Reference

A JSON data parser.

### Enumerations

- enum JsonToken {
  JsonTokenNone = 0, JsonTokenCurlyOpen = 1, JsonTokenCurlyClose = 2, JsonTokenSquaredOpen = 3, JsonTokenSquaredClose = 4, JsonTokenColon = 5, JsonTokenComma = 6, JsonTokenString = 7, JsonTokenNumber = 8, JsonTokenTrue = 9, JsonTokenFalse = 10, JsonTokenNull = 11 }

### Functions

- QVariant parse (const QString &json)
- QVariant parse (const QString &json, bool &success)
- QByteArray serialize (const QVariant &data)
- QByteArray serialize (const QVariant &data, bool &success)

### 6.11.1 Detailed Description

A JSON data parser. Json parses a JSON data into a QVariant hierarchy.

### 6.11.2 Enumeration Type Documentation

#### 6.11.2.1 enum QtJson::JsonToken

**Enumerator**

***JsonTokenNone***

***JsonTokenCurlyOpen***

***JsonTokenCurlyClose***

***JsonTokenSquaredOpen***

***JsonTokenSquaredClose***

***JsonTokenColon***

***JsonTokenComma***

***JsonTokenString***

***JsonTokenNumber***

***JsonTokenTrue***

***JsonTokenFalse***

***JsonTokenNull***

Definition at line 224 of file json.cpp.

### 6.11.3 Function Documentation

#### 6.11.3.1 QVariant QtJson::parse ( const QString & *json* )

parse

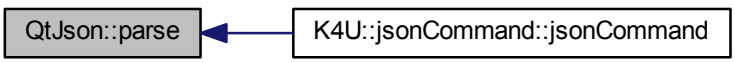
Parse a JSON string

**Parameters**

| | |
|---|---|
| *json* | The JSON data |

Definition at line 60 of file json.cpp.

Here is the caller graph for this function:



#### 6.11.3.2 QVariant QtJson::parse ( const QString & *json*, bool & *success* )

parse

Parse a JSON string

**Parameters**

| | |
|---|---|
| *json* | The JSON data |
| *success* | The success of the parsing |

Definition at line 69 of file json.cpp.

#### 6.11.3.3 QByteArray QtJson::serialize ( const QVariant & *data* )

This method generates a textual JSON representation

**Parameters**

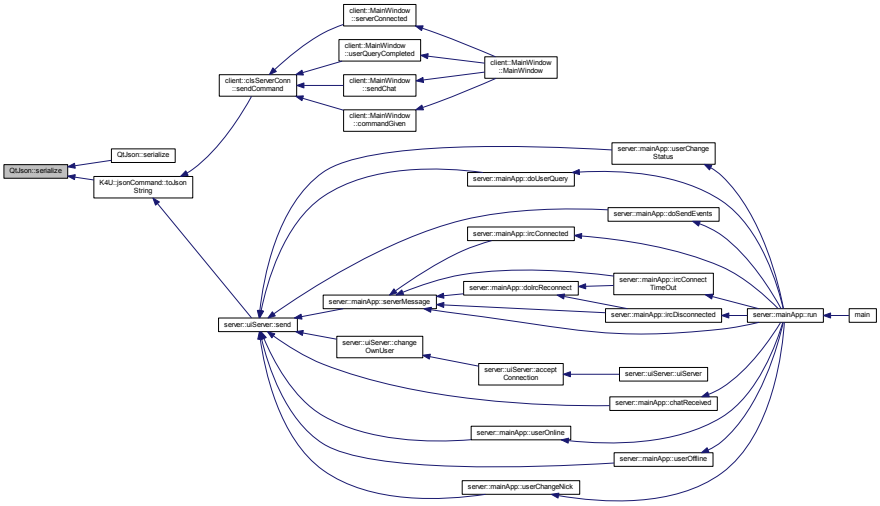
| | |
|---|---|
| *data* | The JSON data generated by the parser. |

**Returns**

QByteArray Textual JSON representation

Definition at line 93 of file json.cpp.

Here is the caller graph for this function:




#### 6.11.3.4 QByteArray QtJson::serialize ( const QVariant & *data*, bool & *success* )

This method generates a textual JSON representation

**Parameters**

| | |
|---|---|
| *data* | The JSON data generated by the parser. |
| *success* | The success of the serialization |

**Returns**

QByteArray Textual JSON representation

Definition at line 99 of file json.cpp.

Here is the call graph for this function:

## 6.12 server Namespace Reference

### Classes

- class clsSettings

  *Settings class.*
- struct nickAndStatus

  *The nickAndStatus struct.*
- class ircClient

  *The ircClient class.*
- class ircUser

  *The ircUser class.*
- class ircUserList

  *Contains all the users.*
- class mainApp

  *The main application.*
- class uiClient

  *The uiClient class.*
- class uiServer

  *The uiServer class.*

### Functions

- const string qts (const QString str)

  *Shorthand to convert a QString to a std::string.*
- nickAndStatus getNickAndStatus (const QString dNick)

  *Converts a nickname to a nickAndStatus (nick|status)*
- const QString getId (const QString fullId)

  *Converts a full IRC-ID to something without the nickname.*
- bool inCommand (const QString ∗command, const char ∗search)

  *Just an instring command. Checks if search is in command.*
- nickAndStatus getNickAndStatusFromId (const QString id)

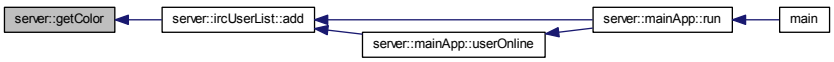
  *Converts an IRC-ID to a nickAndStatus (nick|status)*
- clsLog & operator<< (clsLog &log, const ircUser ∗user)
- ircUser::userColor getColor (int length)

### 6.12.1 Function Documentation

#### 6.12.1.1 ircUser::userColor server::getColor ( int *length* )

Definition at line 181 of file ircuser.cpp.

Here is the caller graph for this function:




#### 6.12.1.2 const QString server::getId ( const QString *fullId* )

Converts a full IRC-ID to something without the nickname.

**Parameters**

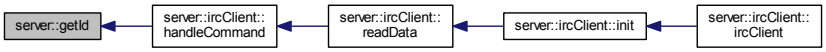
| | |
|---|---|
| *fullId* | A complete IRC-ID (nick\|status!name) |

**Returns**

An IRC-ID usefull to us

Definition at line 37 of file functions.cpp.

Here is the caller graph for this function:




#### 6.12.1.3 nickAndStatus server::getNickAndStatus ( const QString *dNick* )

Converts a nickname to a nickAndStatus (nick|status)

**Parameters**

| | |
|---|---|
| *dNick* | The nickname to convert |

**Returns**

A nickAndStatus object containing the nickname and status of the user

Definition at line 18 of file functions.cpp.

Here is the caller graph for this function:

#### 6.12.1.4 nickAndStatus server::getNickAndStatusFromId ( const QString *id* )

Converts an IRC-ID to a nickAndStatus (nick|status)

**Parameters**

| | |
|---|---|
| *id* | The IRC-ID to convert |

**Returns**

A nickAndStatus object containing the nickname and status of the user

Definition at line 46 of file functions.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

#### 6.12.1.5 bool server::inCommand ( const QString * *command*, const char * *search* )

Just an instring command. Checks if search is in command.

**Parameters**

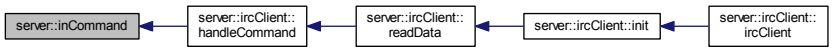
| | |
|---|---|
| *command* | The haystack |
| *search* | The needle |

**Returns**

True when needle is found.

Definition at line 41 of file functions.cpp.

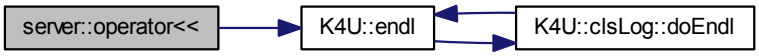
Here is the caller graph for this function:




#### 6.12.1.6 clsLog& server::operator<< ( clsLog & *log*, const ircUser * *user* )

Definition at line 170 of file ircuser.cpp.

Here is the call graph for this function:




#### 6.12.1.7 const std::string server::qts ( const QString *str* )

Shorthand to convert a QString to a std::string.

**Parameters**

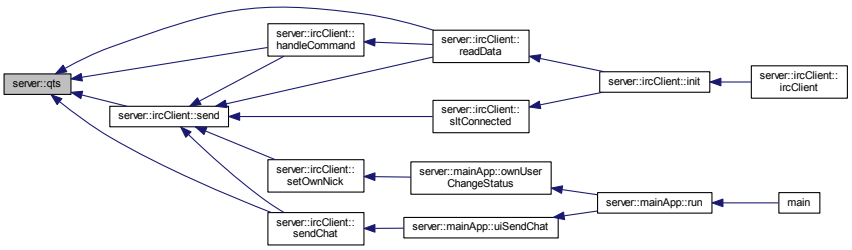
| | |
|---|---|
| *str* | The QString to convert |

**Returns**

A std::string containing str

Definition at line 14 of file functions.cpp.

Here is the caller graph for this function:



## 6.13 shared Namespace Reference

### Namespaces

- namespace events

### Enumerations

- enum level { SERVERMESSAGE_LEVEL_ERROR = 0x01, SERVERMESSAGE_LEVEL_INFO }

### 6.13.1 Enumeration Type Documentation

#### 6.13.1.1 enum shared::level

**Enumerator**

***SERVERMESSAGE_LEVEL_ERROR***

***SERVERMESSAGE_LEVEL_INFO***

Definition at line 6 of file eventservermessage.h.

## 6.14 shared::events Namespace Reference

### Classes

- class clsEvent

  *The main event class.*
- class eventChat

  *Class containing a chatEvent.*
- class eventServerMessage

  *Class that contains data for when the server wants to say something.*
- class eventUserChangeNick

  *Class containing a user who changes his nick-event (nick|status)*
- class eventUserChangeStatus

  *Class containing a user who changes his status-event (nick|status)*
- class eventUserJoin

  *Class containing a user JOIN event.*
- class eventUserLeave

  *Class containing a user PART event.*

### Enumerations

- enum eventTypes {
  EVENTTYPE_NONE = 0x00, EVENTTYPE_CHAT, EVENTTYPE_USER_LEFT, EVENTTYPE_USER_STATUS,
  EVENTTYPE_USER_JOIN, EVENTTYPE_USER_NICK, EVENTTYPE_SERVERMSG }

  *The eventTypes enum.*

### 6.14.1 Enumeration Type Documentation

#### 6.14.1.1 enum shared::events::eventTypes

The eventTypes enum.

**Enumerator**

***EVENTTYPE_NONE*** When all you need is an empty event

***EVENTTYPE_CHAT*** For when a chatmessage has been received

***EVENTTYPE_USER_LEFT*** For when a user left the channel

***EVENTTYPE_USER_STATUS*** For when a user updated his or her status

***EVENTTYPE_USER_JOIN*** For when a user joins the channel

***EVENTTYPE_USER_NICK*** For when a user updates his or her nick

***EVENTTYPE_SERVERMSG*** For when the server wants to tell something to the client

Definition at line 17 of file clsevent.h.

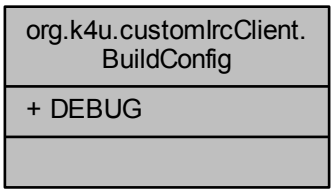
## 6.15 Ui Namespace Reference

# Chapter 7

# Class Documentation

## 7.1 org.k4u.customIrcClient.BuildConfig Class Reference

Collaboration diagram for org.k4u.customIrcClient.BuildConfig:




### Static Public Attributes

- static final boolean DEBUG = true

### 7.1.1 Detailed Description

Definition at line 4 of file BuildConfig.java.

### 7.1.2 Member Data Documentation

#### 7.1.2.1 final boolean org.k4u.customIrcClient.BuildConfig.DEBUG = true [static]

Definition at line 5 of file BuildConfig.java.

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

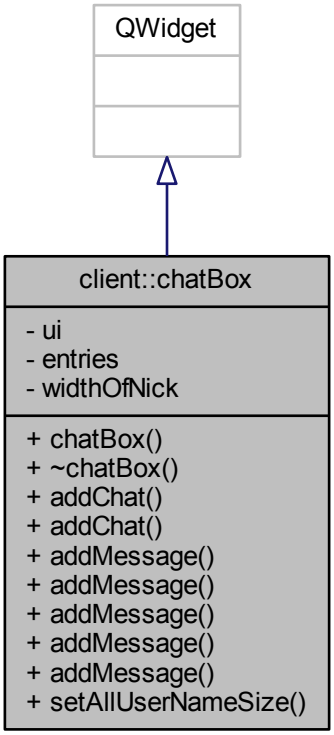
- C++/customIrcClient/android/gen/org/k4u/customIrcClient/BuildConfig.java

## 7.2 client::chatBox Class Reference

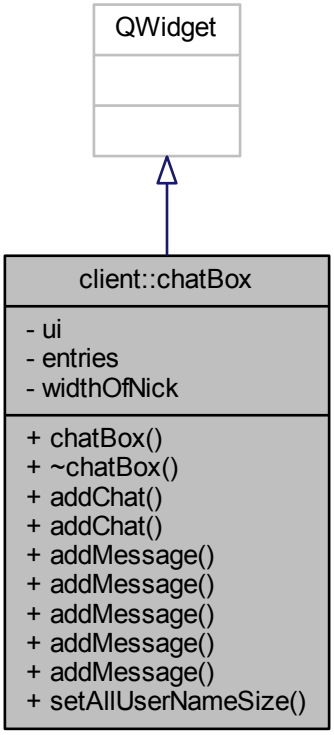
UI Segment containing the chatbox.

`#include <chatbox.h>`

Inheritance diagram for client::chatBox:



Collaboration diagram for client::chatBox:




## Classes

- struct content

## Public Member Functions

- chatBox (QWidget *parent=0)
- ~chatBox ()
- void addChat (ircUser *user, QString message)

  *Adds a chat message to the box, Identified only by a user and a message.*
- void addChat (ircUser *user, QString message, QTime timeOfMessage)

  *Adds a chat message to the box, Identified by a user, a message and a time.*
- void addMessage (QString &message)

  *Adds a notification to the chatbox.*
- void addMessage (const QString &message)

  *Adds a notification to the chatbox.*
- void addMessage (const QString &message, QTime timeOfMessage)

  *Adds a notification to the chatbo with timex.*
- void addMessage (QString &message, QTime timeOfMessage)

  *Adds a notification to the chatbox with time.*
- void addMessage (QString &message, QTime timeOfMessage, QColor &myColor)

*Adds a notification to the chatbox with time and color.*

- void setAllUserNameSize (int newSize)

  *Sets all the user name labels to the same size.*

### Private Types

- enum widgetTypes { WIDGETTYPES_CHATENTRY = 0x00, WIDGETTYPES_NOTIFICATION }

### Private Attributes

- Ui::chatBox * ui
- QVector< content * > entries
- int widthOfNick

## 7.2.1 Detailed Description

UI Segment containing the chatbox.

**Author**

Koen Beckers (K-4U)

Definition at line 23 of file chatbox.h.

## 7.2.2 Member Enumeration Documentation

### 7.2.2.1 enum client::chatBox::widgetTypes [private]

**Enumerator**

***WIDGETTYPES_CHATENTRY***

***WIDGETTYPES_NOTIFICATION***

Definition at line 26 of file chatbox.h.

## 7.2.3 Constructor & Destructor Documentation

### 7.2.3.1 client::chatBox::chatBox ( QWidget * *parent* = 0 ) [explicit]

Definition at line 7 of file chatbox.cpp.

### 7.2.3.2 client::chatBox::~chatBox ( )

Definition at line 17 of file chatbox.cpp.

## 7.2.4 Member Function Documentation

### 7.2.4.1 void client::chatBox::addChat ( ircUser * *user*, QString *message* )

Adds a chat message to the box, Identified only by a user and a message.

**Parameters**

| | |
|---|---|
| *user* | The user that sent the chatmessage |
| *message* | The chat message itself |

Definition at line 22 of file chatbox.cpp.

#### 7.2.4.2 void client::chatBox::addChat ( ircUser * *user*, QString *message*, QTime *timeOfMessage* )

Adds a chat message to the box, Identified by a user, a message and a time.

**Parameters**

| | |
|---|---|
| *user* | The user that sent the chatmessage |
| *message* | The chat message itself |
| *timeOfMessage* | The time on which the message was sent |

Definition at line 31 of file chatbox.cpp.

#### 7.2.4.3 void client::chatBox::addMessage ( QString & *message* )

Adds a notification to the chatbox.

**Parameters**

| | |
|---|---|
| *message* | The notification to add |

Definition at line 40 of file chatbox.cpp.

Here is the caller graph for this function:

#### 7.2.4.4 void client::chatBox::addMessage ( const QString & *message* )

Adds a notification to the chatbox.

**Parameters**

| | |
|---|---|
| *message* | The notification to add |

Definition at line 44 of file chatbox.cpp.

Here is the call graph for this function:

#### 7.2.4.5 void client::chatBox::addMessage ( const QString & *message*, QTime *timeOfMessage* )

Adds a notification to the chatbo with timex.

**Parameters**

| | |
|---|---|
| *message* | The notification to add |
| *timeOfMessage* | The time on which the notification has occured |

Definition at line 49 of file chatbox.cpp.

Here is the call graph for this function:

#### 7.2.4.6 void client::chatBox::addMessage ( QString & *message*, QTime *timeOfMessage* )

Adds a notification to the chatbox with time.

**Parameters**

| | |
|---|---|
| *message* | The notification to be shown |
| *timeOfMessage* | The time on which the notification has occured |

Definition at line 54 of file chatbox.cpp.

Here is the call graph for this function:

#### 7.2.4.7 void client::chatBox::addMessage ( QString & *message*, QTime *timeOfMessage*, QColor & *myColor* )

Adds a notification to the chatbox with time and color.

**Parameters**

| | |
|---|---|
| *message* | The notification to be shown |
| *timeOfMessage* | The time on which the notification has occured |
| *myColor* | The color of the message |

Definition at line 59 of file chatbox.cpp.

#### 7.2.4.8 void client::chatBox::setAllUserNameSize ( int *newSize* )

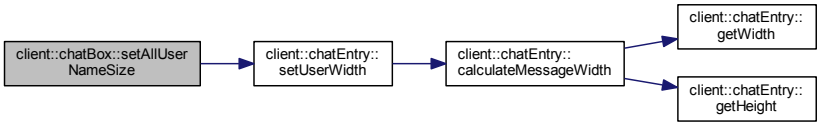
Sets all the user name labels to the same size.

**Parameters**

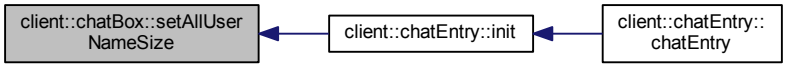
| | |
|---|---|
| *newSize* | The new size This is called from a chatEntry, when it's new calculated width would be too large |

Definition at line 69 of file chatbox.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




### 7.2.5 Member Data Documentation

#### 7.2.5.1 QVector<content*> client::chatBox::entries [private]

Definition at line 97 of file chatbox.h.

#### 7.2.5.2 Ui::chatBox* client::chatBox::ui [private]

Definition at line 96 of file chatbox.h.

#### 7.2.5.3 int client::chatBox::widthOfNick [private]

Definition at line 98 of file chatbox.h.

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

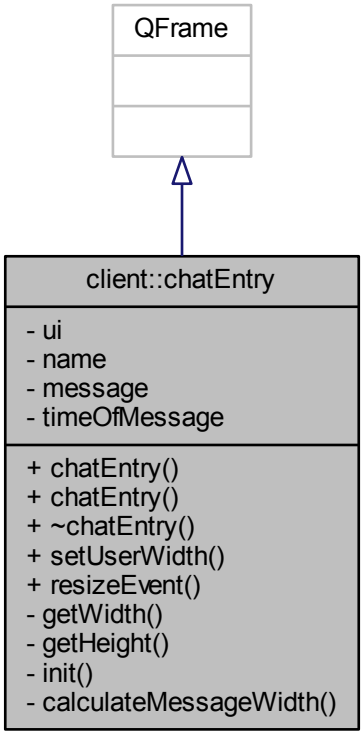
- C++/customIrcClient/chatbox.h
- C++/customIrcClient/chatbox.cpp

## 7.3 client::chatEntry Class Reference

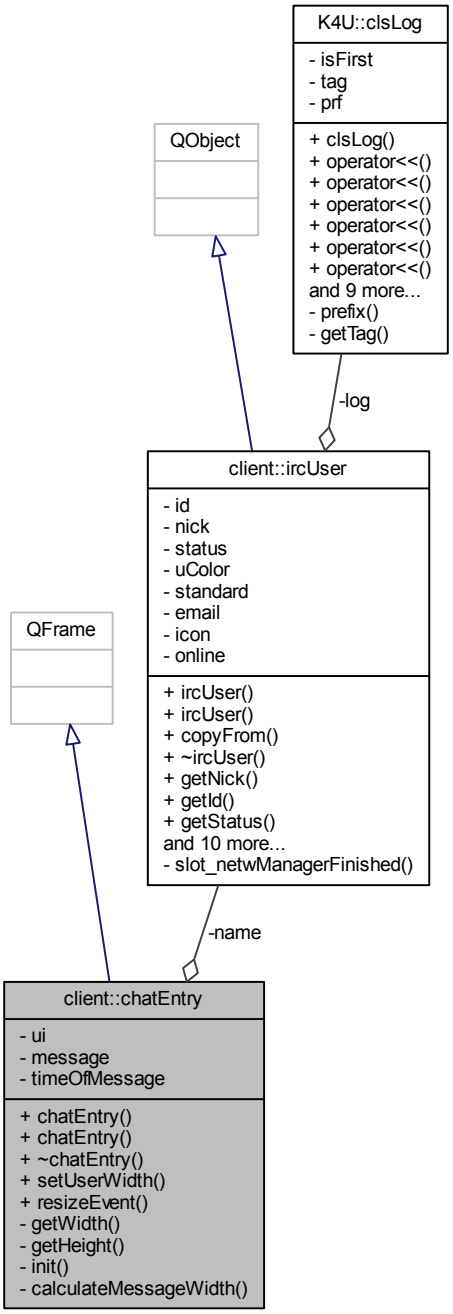
Class that holds a chatmessage for UI.

`#include <chatentry.h>`

Inheritance diagram for client::chatEntry:



Collaboration diagram for client::chatEntry:




## Public Member Functions

- chatEntry (ircUser ∗user, QString message, int width, QWidget ∗parent=0)

  *Constructor not taking a time.*
- chatEntry (ircUser ∗user, QString message, int width, QTime timeOfMessage, QWidget ∗parent=0)

  *Constructor taking a time as argument.*
- ∼chatEntry ()

- void setUserWidth (int newWidth)

  *Used for setting the width of the nick label after initializing.*
- void resizeEvent (QResizeEvent ∗event)

### Private Member Functions

- int getWidth (QLabel ∗l) const
- int getHeight (QLabel ∗l) const
- void init (QWidget ∗parent, int width)
- void calculateMessageWidth ()

### Private Attributes

- Ui::chatEntry ∗ ui
- ircUser ∗ name
- QString message
- QTime timeOfMessage

## 7.3.1 Detailed Description

Class that holds a chatmessage for UI.

**Author**

Koen Beckers (K-4U)

Definition at line 18 of file chatentry.h.

## 7.3.2 Constructor & Destructor Documentation

### 7.3.2.1 client::chatEntry::chatEntry ( ircUser ∗ *user*, QString *message*, int *width*, QWidget ∗ *parent* = 0 ) [explicit]

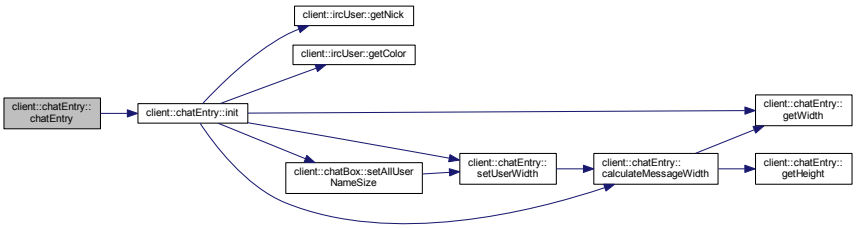
Constructor not taking a time.

**Parameters**

| | |
|---|---|
| *user* | The user that sent the message |
| *message* | The chat message itself |
| *width* | The width of the nick label |
| *parent* | The parent widget |

Definition at line 9 of file chatentry.cpp.

Here is the call graph for this function:




#### 7.3.2.2 client::chatEntry::chatEntry ( ircUser * *user*, QString *message*, int *width*, QTime *timeOfMessage*, QWidget * *parent* = 0 ) [explicit]

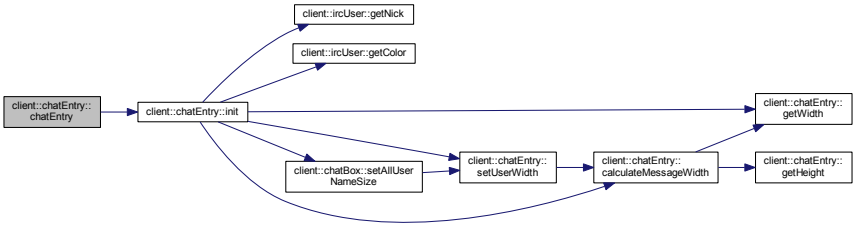
Constructor taking a time as argument.

**Parameters**

| | |
|---|---|
| *user* | The user that sent the message |
| *message* | The chat message itself |
| *width* | The width of the nick label |
| *timeOfMessage* | The time at which the message was sent |
| *parent* | The parent widget |

Definition at line 41 of file chatentry.cpp.

Here is the call graph for this function:




#### 7.3.2.3 client::chatEntry::~chatEntry ( )

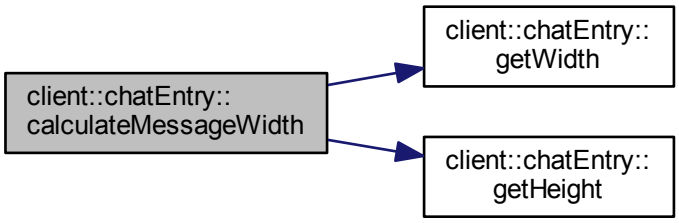
Definition at line 74 of file chatentry.cpp.

### 7.3.3 Member Function Documentation

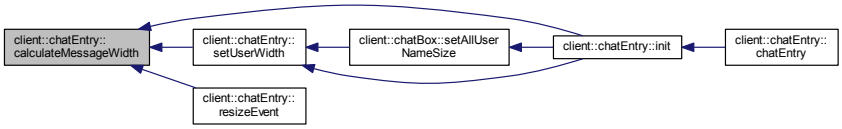
#### 7.3.3.1 void client::chatEntry::calculateMessageWidth ( ) [private]

Definition at line 51 of file chatentry.cpp.

Here is the call graph for this function:



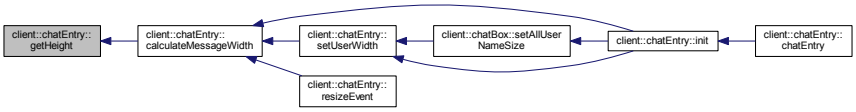

Here is the caller graph for this function:




#### 7.3.3.2 int client::chatEntry::getHeight ( QLabel * *l* ) const [private]

Definition at line 70 of file chatentry.cpp.

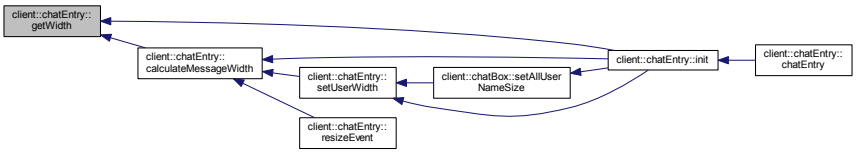
Here is the caller graph for this function:




#### 7.3.3.3 int client::chatEntry::getWidth ( QLabel * *l* ) const [private]

Definition at line 66 of file chatentry.cpp.

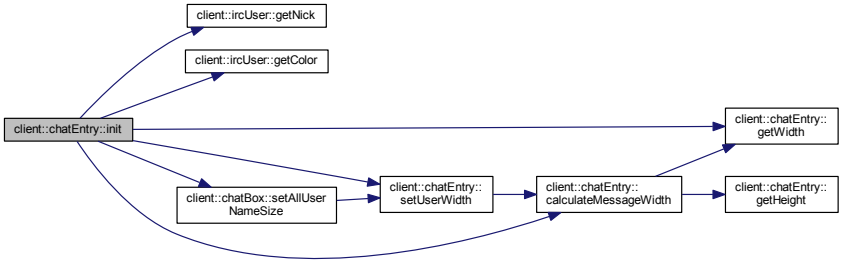
Here is the caller graph for this function:



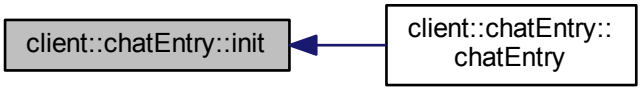
#### 7.3.3.4 void client::chatEntry::init ( QWidget * *parent*, int *width* ) [private]

Definition at line 19 of file chatentry.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.3.3.5 void client::chatEntry::resizeEvent ( QResizeEvent * *event* )

Definition at line 85 of file chatentry.cpp.

Here is the call graph for this function:

#### 7.3.3.6 void client::chatEntry::setUserWidth ( int *newWidth* )

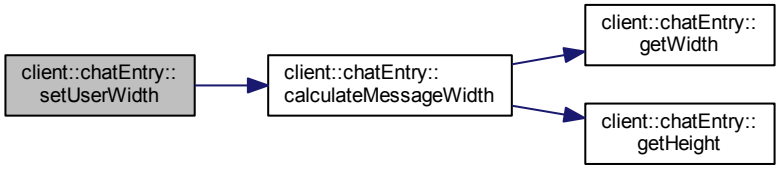
Used for setting the width of the nick label after initializing.

**Parameters**

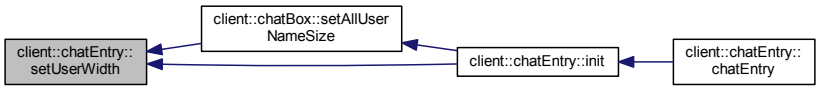
| | |
|---|---|
| *newWidth* | The new width |

Definition at line 78 of file chatentry.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




### 7.3.4 Member Data Documentation

#### 7.3.4.1 QString client::chatEntry::message [private]

Definition at line 56 of file chatentry.h.

#### 7.3.4.2 ircUser* client::chatEntry::name [private]

Definition at line 55 of file chatentry.h.

#### 7.3.4.3 QTime client::chatEntry::timeOfMessage [private]

Definition at line 62 of file chatentry.h.

#### 7.3.4.4 Ui::chatEntry* client::chatEntry::ui [private]

Definition at line 54 of file chatentry.h.

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

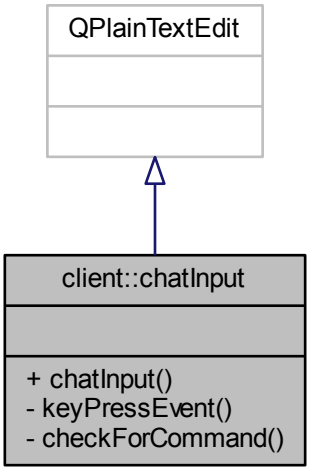
- C++/customIrcClient/chatentry.h
- C++/customIrcClient/chatentry.cpp

## 7.4 client::chatInput Class Reference

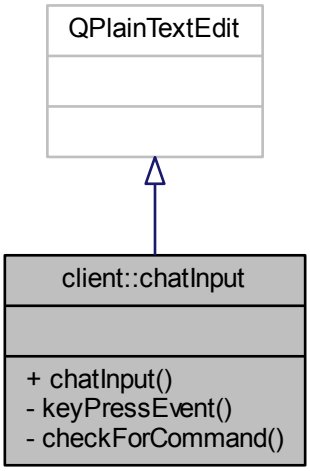
UI segment containing the chat input.

```
#include <chatinput.h>
```

Inheritance diagram for client::chatInput:




Collaboration diagram for client::chatInput:



### Signals

- void sendChat (QString message)

  *Emitted when the user presses enter and the chatbox does not contain a command.*
- void commandGiven (QString command, QList< QString > args)

  *Emitted when the users presses enter and the chatbox contains a command.*

### Public Member Functions

- chatInput (QWidget *parent=0)

### Private Member Functions

- void keyPressEvent (QKeyEvent *e)
- void checkForCommand (QString text)

## 7.4.1 Detailed Description

UI segment containing the chat input.

**Author**

Koen Beckers (K-4U)

Definition at line 13 of file chatinput.h.

## 7.4.2 Constructor & Destructor Documentation

### 7.4.2.1 client::chatInput::chatInput ( QWidget * *parent* = 0 ) [explicit]

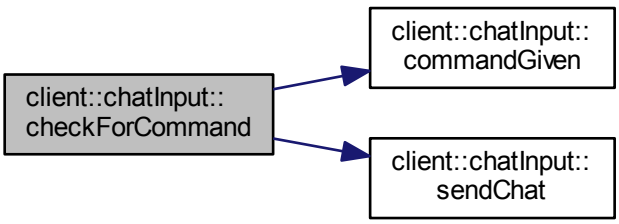
Definition at line 7 of file chatinput.cpp.

## 7.4.3 Member Function Documentation

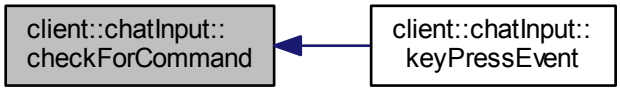
### 7.4.3.1 void client::chatInput::checkForCommand ( QString *text* ) [private]

Definition at line 31 of file chatinput.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:




#### 7.4.3.2 void client::chatInput::commandGiven ( QString *command*, QList< QString > *args* ) [signal]

Emitted when the users presses enter and the chatbox contains a command.

**Parameters**

| | |
|---|---|
| *command* | The command send |
| *args* | A QList<QString> of arguments (split at space) |

Here is the caller graph for this function:

#### 7.4.3.3 void client::chatInput::keyPressEvent ( QKeyEvent * *e* ) [private]

Definition at line 13 of file chatinput.cpp.

Here is the call graph for this function:

#### 7.4.3.4 void client::chatInput::sendChat ( QString *message* ) [signal]

Emitted when the user presses enter and the chatbox does not contain a command.

**Parameters**

| | |
|---|---|
| *message* | The message to be sent |

Here is the caller graph for this function:

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

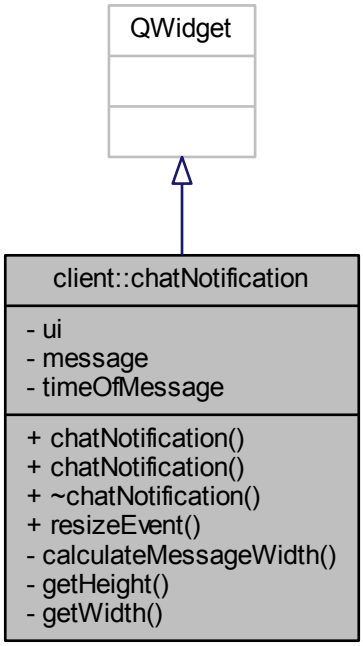
- C++/customIrcClient/chatinput.h
- C++/customIrcClient/chatinput.cpp

## 7.5 client::chatNotification Class Reference

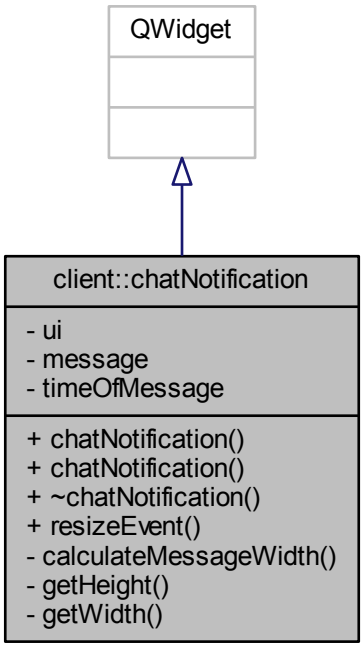
UI Segment containing a notification in the chatbox.

`#include <chatnotification.h>`

Inheritance diagram for client::chatNotification:



Collaboration diagram for client::chatNotification:




### Public Member Functions

- chatNotification (QString message, QWidget ∗parent=0)

  *Constructor taking a message.*
- chatNotification (QString message, QTime timeOfMessage, QColor myColor, QWidget ∗parent=0)

  *Constructor taking a message and a time.*
- ∼chatNotification ()
- void resizeEvent (QResizeEvent ∗event)

### Private Member Functions

- void calculateMessageWidth ()

  *Calculates the message width to detirmine it should be wrapped.*
- int getHeight (QLabel ∗l) const
- int getWidth (QLabel ∗l) const

### Private Attributes

- Ui::chatNotification ∗ ui
- QString message
- QTime timeOfMessage

### 7.5.1 Detailed Description

UI Segment containing a notification in the chatbox.

**Author**

Koen Beckers (K-4U)

Definition at line 17 of file chatnotification.h.

### 7.5.2 Constructor & Destructor Documentation

#### 7.5.2.1 client::chatNotification::chatNotification ( QString *message*, QWidget * *parent* = 0 ) [explicit]

Constructor taking a message.

**Parameters**

| | |
|---|---|
| *message* | The message to be shown |
| *parent* | The parent widget |

Definition at line 8 of file chatnotification.cpp.

Here is the call graph for this function:

#### 7.5.2.2 client::chatNotification::chatNotification ( QString *message*, QTime *timeOfMessage*, QColor *myColor*, QWidget * *parent* = 0 ) [explicit]

Constructor taking a message and a time.

**Parameters**

| | |
|---|---|
| *message* | The message to be shown |
| *timeOfMessage* | The time on which the message has occured |
| *myColor* | The color of the message |
| *parent* | The parent widget |

Definition at line 24 of file chatnotification.cpp.

Here is the call graph for this function:

#### 7.5.2.3 client::chatNotification::~chatNotification ( )

Definition at line 44 of file chatnotification.cpp.

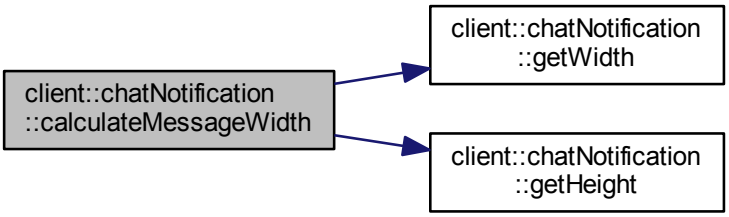
### 7.5.3 Member Function Documentation

#### 7.5.3.1 void client::chatNotification::calculateMessageWidth ( ) [private]

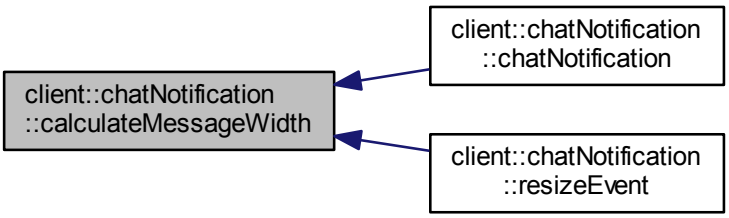
Calculates the message width to detirmine it should be wrapped.

Definition at line 49 of file chatnotification.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:




#### 7.5.3.2 int client::chatNotification::getHeight ( QLabel * *l* ) const [private]

Definition at line 66 of file chatnotification.cpp.

Here is the caller graph for this function:

#### 7.5.3.3 int client::chatNotification::getWidth ( QLabel * *l* ) const [private]

Definition at line 62 of file chatnotification.cpp.

Here is the caller graph for this function:

#### 7.5.3.4 void client::chatNotification::resizeEvent ( QResizeEvent * *event* )

Definition at line 72 of file chatnotification.cpp.

Here is the call graph for this function:

### 7.5.4 Member Data Documentation

#### 7.5.4.1 QString client::chatNotification::message [private]

Definition at line 50 of file chatnotification.h.

#### 7.5.4.2 QTime client::chatNotification::timeOfMessage [private]

Definition at line 51 of file chatnotification.h.

#### 7.5.4.3 Ui::chatNotification* client::chatNotification::ui [private]

Definition at line 49 of file chatnotification.h.

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

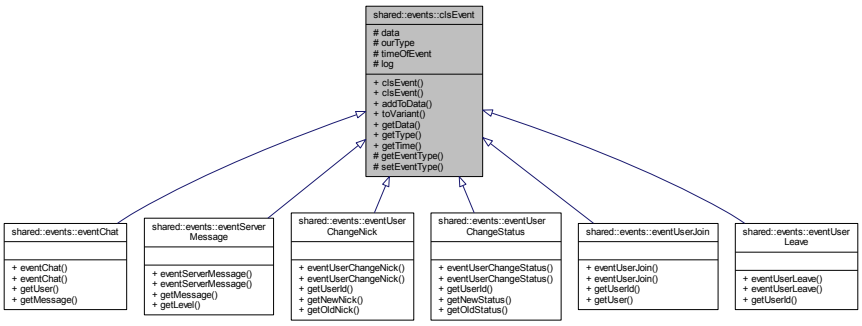
- C++/customIrcClient/chatnotification.h
- C++/customIrcClient/chatnotification.cpp

## 7.6 shared::events::clsEvent Class Reference

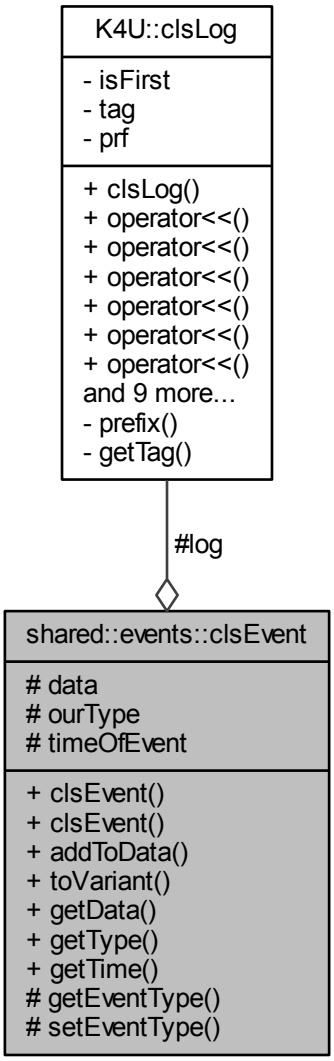
The main event class.

`#include <clsevent.h>`

Inheritance diagram for shared::events::clsEvent:



Collaboration diagram for shared::events::clsEvent:




## Public Member Functions

- clsEvent ()

  *Empty initializer function.*
- clsEvent (QVariantMap data)

  *Initializer function that accepts data immediately.*
- void addToData (const QString &key, const QVariant data)

  *Adds data to the events data map.*
- QVariant toVariant () const

  *Converts the current data to a QVariant for sending with JSON.*
- QVariant getData (QString key) const

  *Get data from the events data map.*

- eventTypes getType () const

  *Gets this events type.*
- QTime getTime () const

  *Gets the time on which the event was created.*

### Protected Member Functions

- QString getEventType () const

  *Gets the event type as a string.*
- void setEventType (const QString eventString)

  *Sets the event type as a string.*

### Protected Attributes

- QVariantMap data

  *The events data map.*
- eventTypes ourType
- QDateTime timeOfEvent
- clsLog log

## 7.6.1 Detailed Description

The main event class.

**Author**

Koen Beckers (K-4U)

Definition at line 32 of file clsevent.h.

## 7.6.2 Constructor & Destructor Documentation

### 7.6.2.1 shared::events::clsEvent::clsEvent ( ) [explicit]

Empty initializer function.

Definition at line 5 of file clsevent.cpp.

### 7.6.2.2 shared::events::clsEvent::clsEvent ( QVariantMap *data* ) [explicit]

Initializer function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 11 of file clsevent.cpp.

Here is the call graph for this function:

### 7.6.3 Member Function Documentation

#### 7.6.3.1 void shared::events::clsEvent::addToData ( const QString & *key*, const QVariant *data* )

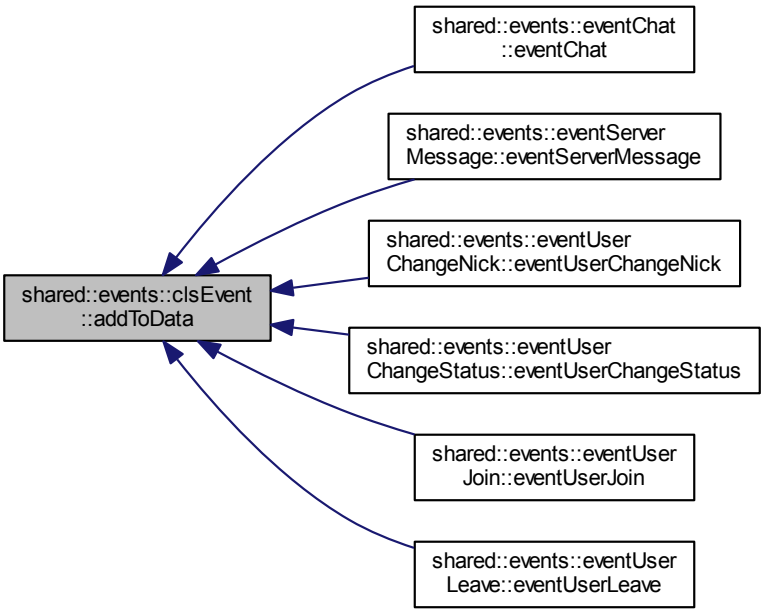
Adds data to the events data map.

**Parameters**

| | |
|---|---|
| *key* | The key to add the data under |
| *data* | The data to add |

Definition at line 19 of file clsevent.cpp.

Here is the caller graph for this function:



#### 7.6.3.2 QVariant shared::events::clsEvent::getData ( QString *key* ) const

Get data from the events data map.

**Parameters**

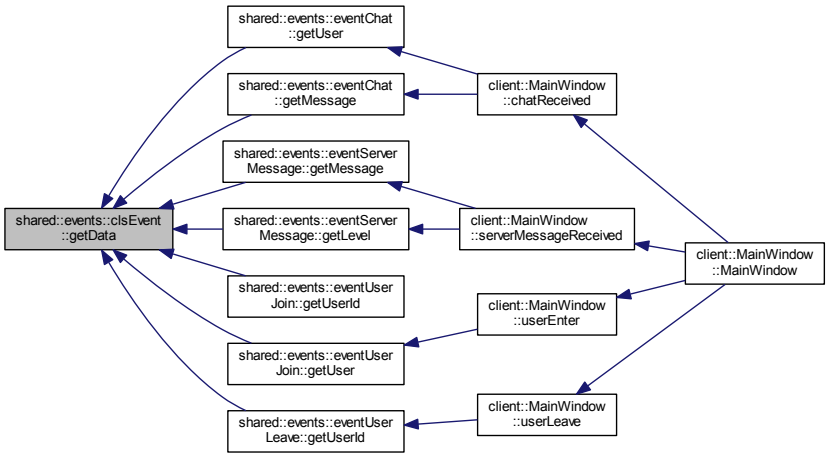
| | |
|---|---|
| *key* | The key to retrieve |

**Returns**

A QVariant containing the data

Definition at line 32 of file clsevent.cpp.

Here is the caller graph for this function:




#### 7.6.3.3 QString shared::events::clsEvent::getEventType ( ) const [protected]

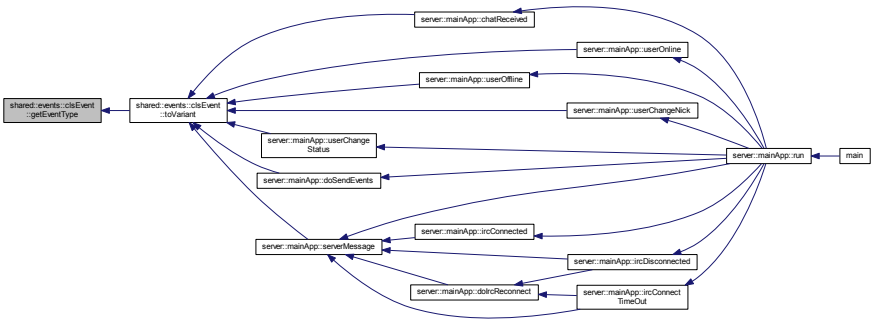
Gets the event type as a string.

**Returns**

The event type as a string

Definition at line 63 of file clsevent.cpp.

Here is the caller graph for this function:



#### 7.6.3.4 QTime shared::events::clsEvent::getTime ( ) const

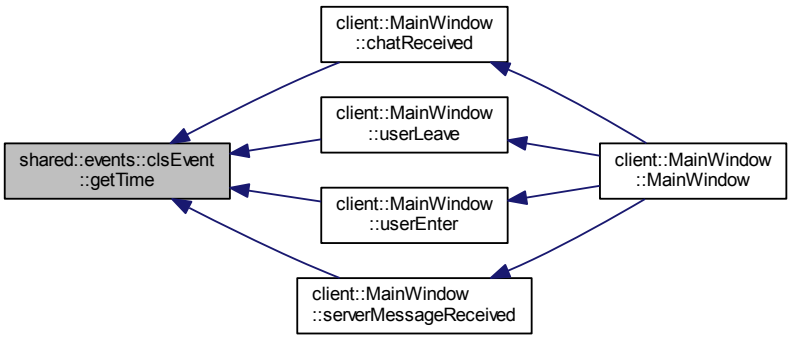
Gets the time on which the event was created.

**Returns**

The time on which the event was created

Definition at line 41 of file clsevent.cpp.

Here is the caller graph for this function:




#### 7.6.3.5 eventTypes shared::events::clsEvent::getType ( ) const

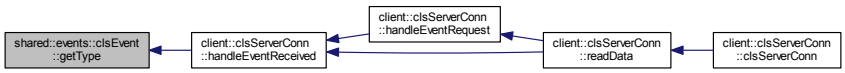
Gets this events type.

**Returns**

An eventTypes object containing the eventType

Definition at line 36 of file clsevent.cpp.

Here is the caller graph for this function:




#### 7.6.3.6 void shared::events::clsEvent::setEventType ( const QString *eventString* ) [protected]

Sets the event type as a string.

**Parameters**

| | |
|---|---|
| *eventString* | The eventType as a string |

Definition at line 45 of file clsevent.cpp.

Here is the caller graph for this function:

#### 7.6.3.7 QVariant shared::events::clsEvent::toVariant ( ) const

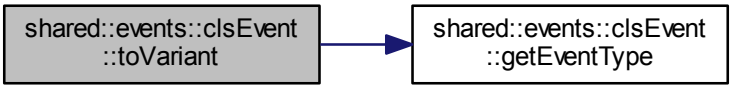
Converts the current data to a QVariant for sending with JSON.

**Returns**

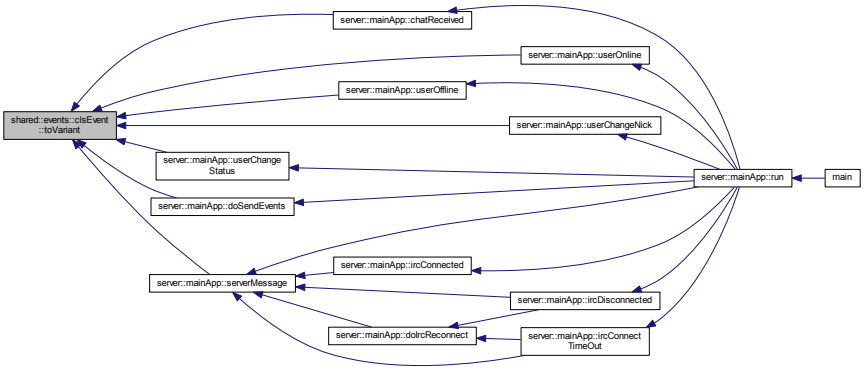
The event as a QVariant object

Definition at line 23 of file clsevent.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




### 7.6.4 Member Data Documentation

#### 7.6.4.1 QVariantMap shared::events::clsEvent::data [protected]

The events data map.

Definition at line 82 of file clsevent.h.

#### 7.6.4.2 clsLog shared::events::clsEvent::log [protected]

Definition at line 98 of file clsevent.h.

#### 7.6.4.3 eventTypes shared::events::clsEvent::ourType [protected]

Definition at line 83 of file clsevent.h.

#### 7.6.4.4 QDateTime shared::events::clsEvent::timeOfEvent [protected]

Definition at line 84 of file clsevent.h.

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

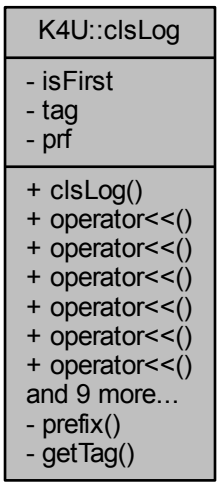
- C++/resources/clsevent.h
- C++/resources/clsevent.cpp

## 7.7 K4U::clsLog Class Reference

Custom logging class.

`#include <clslog.h>`

Collaboration diagram for K4U::clsLog:



### Public Member Functions

- clsLog (logTags tag)

  *clsLog constructor*
- clsLog & operator<< (clsLog &(∗manip)(clsLog &))

  *Used for end line.*
- clsLog & operator<< (QString msg)
- clsLog & operator<< (const QString ∗msg)
- clsLog & operator<< (const char ∗msg)
- clsLog & operator<< (int msg)
- clsLog & operator<< (clsLog ∗log)
- clsLog & operator<< (bool msg)
- clsLog & operator<< (QVariantMap &map)
- clsLog & operator<< (const QVariantMap &map)
- clsLog & operator<< (const QVariant &msg)
- clsLog & operator<< (QVariant &msg)
- clsLog & operator<< (QByteArray &bArr)
- clsLog & operator<< (const QByteArray &bArr)
- void doEndl ()
- void setPrefix (QString prefix)

  *Sets a custom prefix, used for when more than one object uses the same log tag.*

### Private Member Functions

- void prefix ()
- std::string getTag ()

### Private Attributes

- bool isFirst
- logTags tag
- QString prf

## 7.7.1 Detailed Description

Custom logging class.

**Author**

Koen Beckers (K-4U) Because std::out doesn't allow for QT-objects, i wrote my own parser on top of this. It also allows tags and date, so that you can allways see where the debug came from

Definition at line 27 of file clslog.h.

## 7.7.2 Constructor & Destructor Documentation

### 7.7.2.1 K4U::clsLog::clsLog ( logTags *tag* )

clsLog constructor

**Parameters**

| | |
|---|---|
| *tag* | The tag to use while printing |

Definition at line 7 of file clslog.cpp.

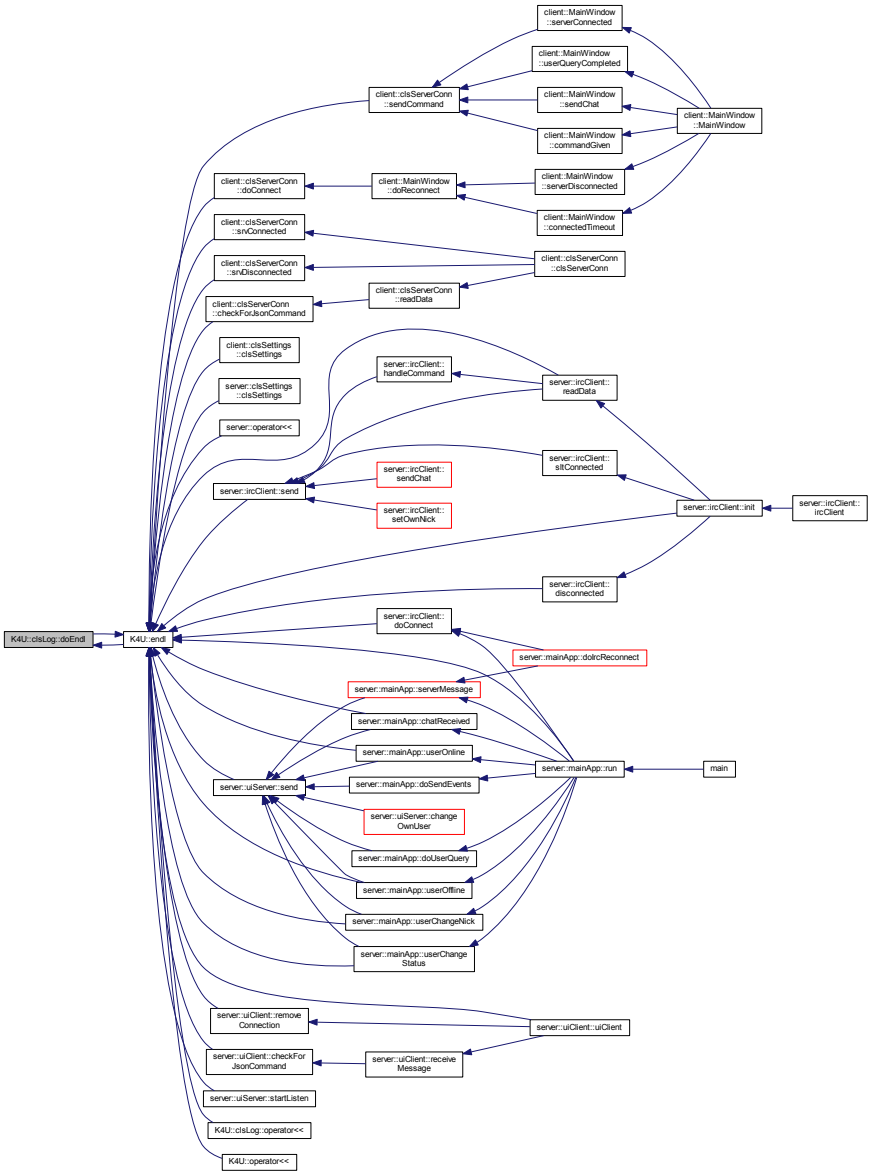
### 7.7.3 Member Function Documentation

#### 7.7.3.1 void K4U::clsLog::doEndl ( )

Definition at line 70 of file clslog.cpp.

Here is the call graph for this function:

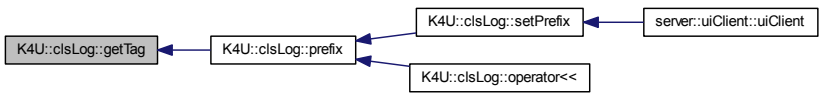
Here is the caller graph for this function:




### 7.7.3.2 std::string K4U::clsLog::getTag ( ) [private]

Definition at line 125 of file clslog.cpp.

Here is the caller graph for this function:



#### 7.7.3.3 clsLog & K4U::clsLog::operator<< ( clsLog &(*)(clsLog &) *manip* )

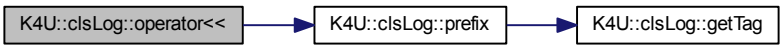
Used for end line.

Definition at line 13 of file clslog.cpp.

#### 7.7.3.4 clsLog & K4U::clsLog::operator<< ( QString *msg* )

Definition at line 96 of file clslog.cpp.

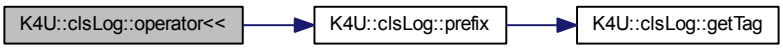
Here is the call graph for this function:




#### 7.7.3.5 clsLog & K4U::clsLog::operator<< ( const QString * *msg* )

Definition at line 103 of file clslog.cpp.

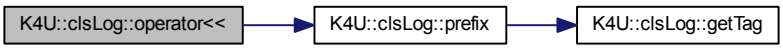
Here is the call graph for this function:




#### 7.7.3.6 clsLog & K4U::clsLog::operator<< ( const char * *msg* )

Definition at line 110 of file clslog.cpp.

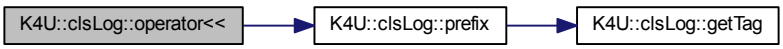
Here is the call graph for this function:




#### 7.7.3.7 clsLog & K4U::clsLog::operator<< ( int *msg* )

Definition at line 117 of file clslog.cpp.

Here is the call graph for this function:




#### 7.7.3.8 clsLog& K4U::clsLog::operator<< ( clsLog * *log* )

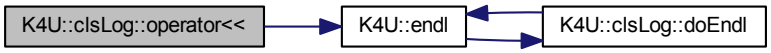
#### 7.7.3.9 clsLog & K4U::clsLog::operator<< ( bool *msg* )

Definition at line 17 of file clslog.cpp.

#### 7.7.3.10 clsLog & K4U::clsLog::operator<< ( QVariantMap & *map* )

Definition at line 26 of file clslog.cpp.

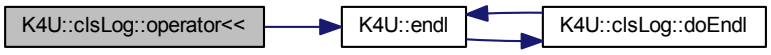
Here is the call graph for this function:




#### 7.7.3.11 clsLog & K4U::clsLog::operator<< ( const QVariantMap & *map* )

Definition at line 34 of file clslog.cpp.

Here is the call graph for this function:




#### 7.7.3.12 clsLog & K4U::clsLog::operator<< ( const QVariant & *msg* )

Definition at line 42 of file clslog.cpp.

#### 7.7.3.13 clsLog & K4U::clsLog::operator<< ( QVariant & *msg* )

Definition at line 48 of file clslog.cpp.

#### 7.7.3.14 clsLog & K4U::clsLog::operator<< ( QByteArray & *bArr* )

Definition at line 54 of file clslog.cpp.

#### 7.7.3.15 clsLog & K4U::clsLog::operator<< ( const QByteArray & *bArr* )

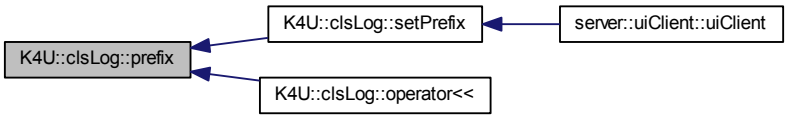
Definition at line 59 of file clslog.cpp.

#### 7.7.3.16 void K4U::clsLog::prefix ( ) [private]

Definition at line 79 of file clslog.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:




#### 7.7.3.17 void K4U::clsLog::setPrefix ( QString *prefix* )

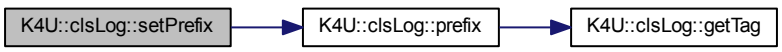
Sets a custom prefix, used for when more than one object uses the same log tag.

**Parameters**

| | |
|---|---|
| *prefix* | The prefix to show |

Definition at line 75 of file clslog.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

### 7.7.4 Member Data Documentation

#### 7.7.4.1 bool K4U::clsLog::isFirst [private]

Definition at line 62 of file clslog.h.

#### 7.7.4.2 QString K4U::clsLog::prf [private]

Definition at line 67 of file clslog.h.

#### 7.7.4.3 logTags K4U::clsLog::tag [private]

Definition at line 63 of file clslog.h.

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

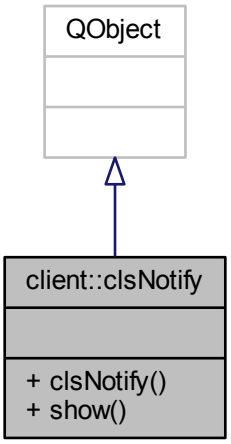
- C++/resources/clslog.h
- C++/resources/clslog.cpp

## 7.8 client::clsNotify Class Reference

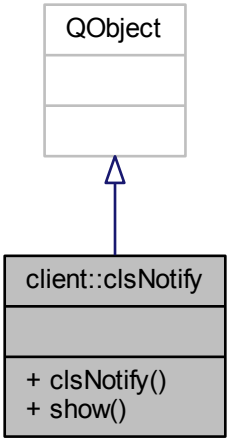
Notification class.

```
#include <clsnotify.h>
```

Inheritance diagram for client::clsNotify:




Collaboration diagram for client::clsNotify:




### Public Member Functions

- clsNotify (const QString title, const QString body, QObject *parent=0)
- void show () const

### 7.8.1 Detailed Description

Notification class.

**Warning**

Does not work yet!

**Author**

Koen Beckers (K-4U)

Definition at line 18 of file clsnotify.h.

### 7.8.2 Constructor & Destructor Documentation

#### 7.8.2.1 client::clsNotify::clsNotify ( const QString *title*, const QString *body*, QObject * *parent* = 0 ) [explicit]

Definition at line 5 of file clsnotify.cpp.

### 7.8.3 Member Function Documentation

#### 7.8.3.1 void client::clsNotify::show ( ) const

Definition at line 16 of file clsnotify.cpp.

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

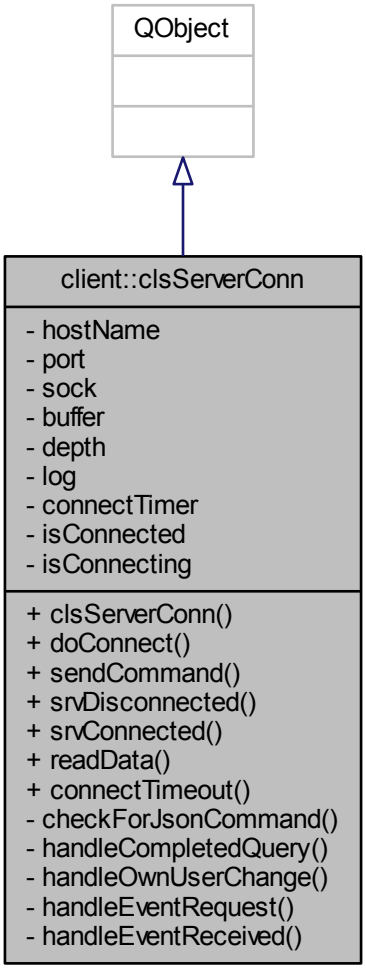
- C++/customIrcClient/desktop/clsnotify.h
- C++/customIrcClient/desktop/clsnotify.cpp

## 7.9 client::clsServerConn Class Reference

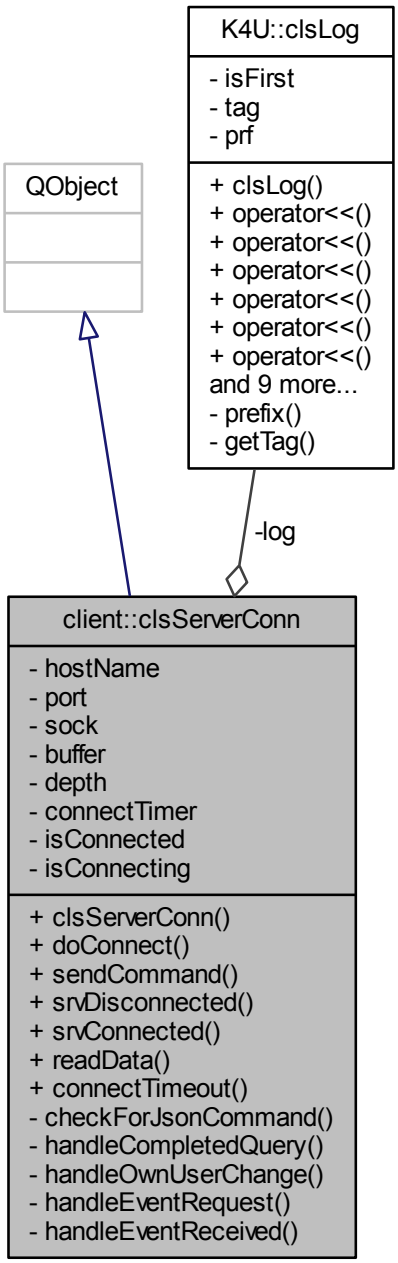
Class that has a connection to the server.

```
#include <clsserverconn.h>
```

Inheritance diagram for client::clsServerConn:



Collaboration diagram for client::clsServerConn:




## Public Slots

- void srvDisconnected ()
- void srvConnected ()
- void readData ()
- void connectTimeout ()

## Signals

- void chatReceived (eventChat *event)

  *Emitted when a chat event has been received.*
- void connected ()

  *Emitted when the client has a succesfull connection to the server.*
- void disconnected ()

  *Emitted when the client lost connection to the server.*
- void userQueryCompleted (QVector< ircUser * > users)

  *Emitted when the server has sent us all the users.*
- void userStatusChange (eventUserChangeStatus *changeEvent)

  *Emitted when a user has changed his status (nick|status)*
- void userChangeNick (eventUserChangeNick *changeEvent)

  *Emitted when a user has changed his nick (nick|status)*
- void userLeave (eventUserLeave *leaveEvent)

  *Triggered when a user left the channel.*
- void userEnter (eventUserJoin *newUser)

  *Triggered when a user enters the channel.*
- void serverMessageReceived (eventServerMessage *theMessage)

  *Triggered when the server wants us to know something.*
- void sgnConnectTimeout ()

  *Triggered when it cannot connect.*

## Public Member Functions

- clsServerConn (const QString hostName, const int port, QObject *parent=0)

  *Constructor taking a hostname and port.*
- void doConnect ()

  *Actually connect.*
- void sendCommand (jsonCommand &toSend)

  *Send a JSON command to the server.*

## Private Member Functions

- bool checkForJsonCommand (QByteArray toAdd, jsonCommand &comm)

  *Adds data to buffer and checks for json commands.*
- void handleCompletedQuery (jsonCommand &comm)

  *The server has given us all the users. Handle it.*
- void handleOwnUserChange (jsonCommand &comm)

  *Another client has changed our own user status.*
- void handleEventRequest (jsonCommand &comm)

  *Event request has been done by us, and data has been returned.*
- void handleEventReceived (QVariantMap data)

  *Event has been received by the client. Parse it.*

### Private Attributes

- QString hostName
- int port
- QTcpSocket * sock
- QString * buffer
- int depth
- clsLog log
- QTimer * connectTimer
- bool isConnected
- bool isConnecting

## 7.9.1 Detailed Description

Class that has a connection to the server.

**Author**

Koen Beckers (K-4U)

Definition at line 27 of file clsserverconn.h.

## 7.9.2 Constructor & Destructor Documentation

### 7.9.2.1 client::clsServerConn::clsServerConn ( const QString *hostName*, const int *port*, QObject * *parent* = 0 ) [explicit]

Constructor taking a hostname and port.

**Parameters**

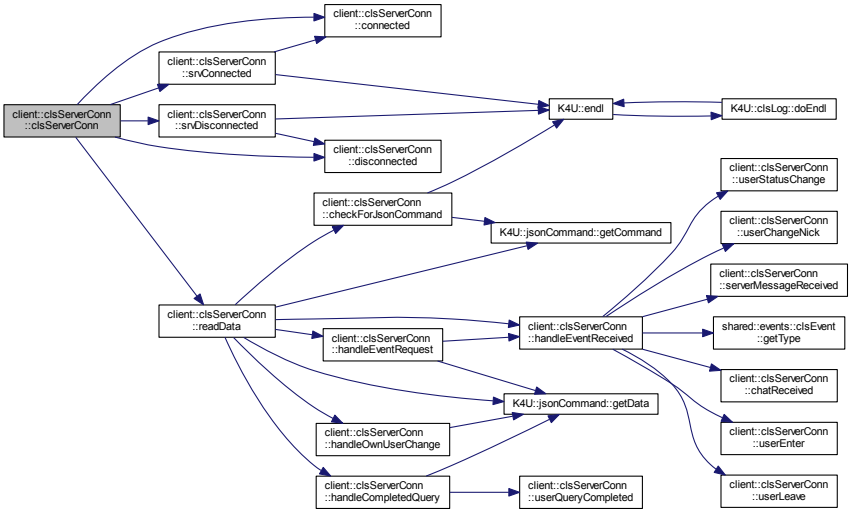
| | |
|---|---|
| *hostName* | The hostname to connect to |
| *port* | The port to connect on |
| *parent* | The parent widget, Not allways used |

**Note**

This only initializes the socket, it does not connect!

Definition at line 6 of file clsserverconn.cpp.

Here is the call graph for this function:




### 7.9.3 Member Function Documentation

#### 7.9.3.1 void client::clsServerConn::chatReceived ( eventChat * *event* ) [signal]

Emitted when a chat event has been received.

**Parameters**

| | |
|---|---|
| *event* | |

Here is the caller graph for this function:

#### 7.9.3.2 bool client::clsServerConn::checkForJsonCommand ( QByteArray *toAdd,* jsonCommand & *comm* ) [private]

Adds data to buffer and checks for json commands.

**Parameters**

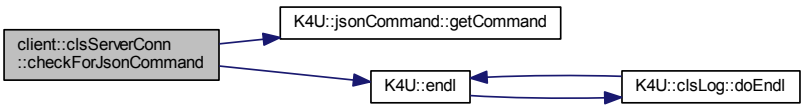
| | |
|---|---|
| *toAdd* | The bytes to add |
| *comm* | Pointer to a jsonCommand to be filled |

**Returns**

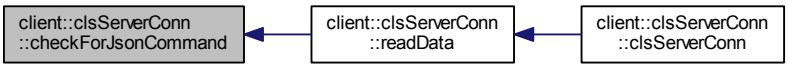
true if a jsonCommand was detected Constantly looks for a JSON command in the buffer This is done with the depth. Every time a { or [ is detected, depth is raised. Every time a } or ] is detected, depth is decreased. When depth is 0, there should be a complete json command!

Definition at line 116 of file clsserverconn.cpp.

Here is the call graph for this function:



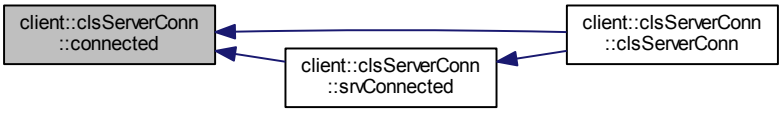

Here is the caller graph for this function:




#### 7.9.3.3 void client::clsServerConn::connected ( ) [signal]

Emitted when the client has a succesfull connection to the server.

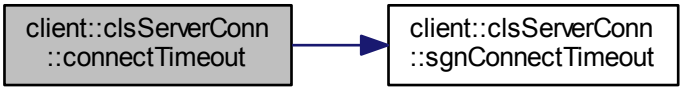
Here is the caller graph for this function:




#### 7.9.3.4 void client::clsServerConn::connectTimeout ( ) [slot]

Definition at line 160 of file clsserverconn.cpp.

Here is the call graph for this function:

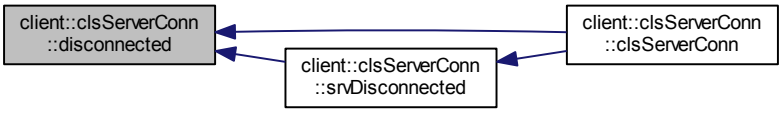


Here is the caller graph for this function:

#### 7.9.3.5 void client::clsServerConn::disconnected ( ) [signal]

Emitted when the client lost connection to the server.

Here is the caller graph for this function:

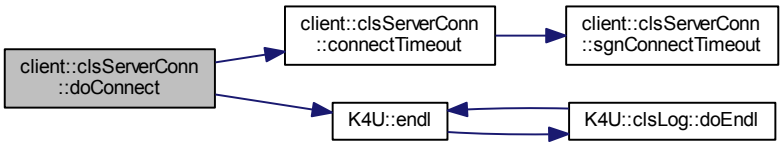



#### 7.9.3.6 void client::clsServerConn::doConnect ( )

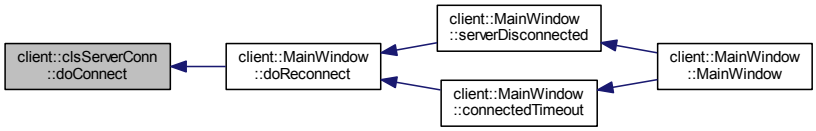
Actually connect.

Definition at line 28 of file clsserverconn.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:



#### 7.9.3.7 void client::clsServerConn::handleCompletedQuery ( jsonCommand & *comm* ) [private]

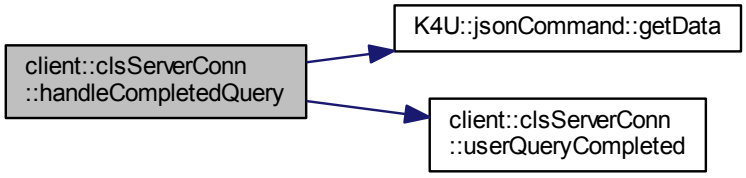
The server has given us all the users. Handle it.

**Parameters**

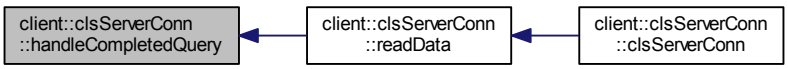
| | |
|---|---|
| *comm* | The jsonCommand containing all the data |

Definition at line 61 of file clsserverconn.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.9.3.8 void client::clsServerConn::handleEventReceived ( QVariantMap *data* ) [private]

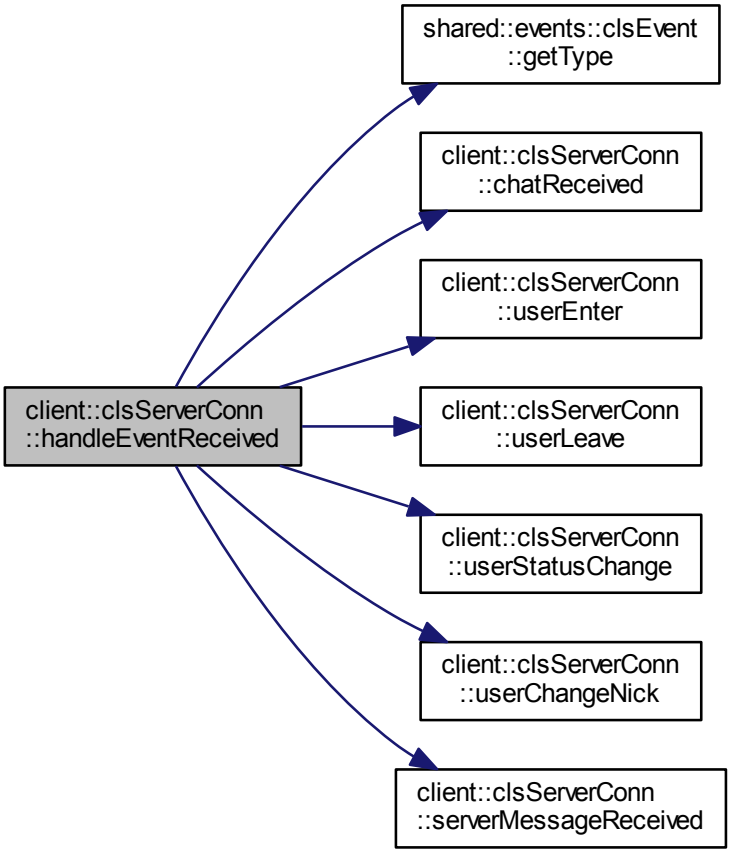
Event has been received by the client. Parse it.

**Parameters**

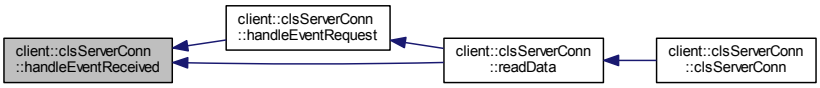
| | |
|---|---|
| *data* | A QVariantMap containing all the data |

Definition at line 88 of file clsserverconn.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.9.3.9 void client::clsServerConn::handleEventRequest ( jsonCommand & *comm* ) [private]

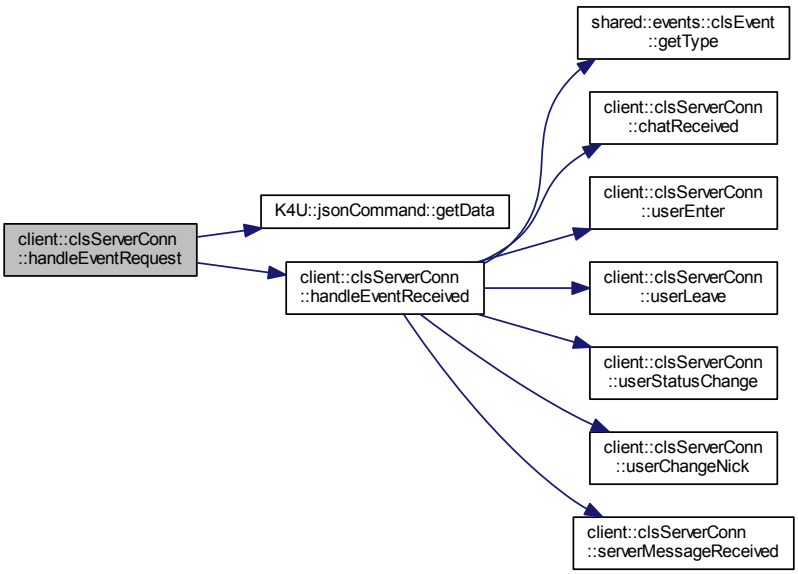
Event request has been done by us, and data has been returned.

**Parameters**

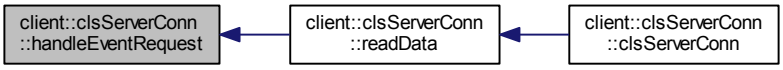
| | |
|---|---|
| *comm* | The jsonCommand containing all the events |

Definition at line 80 of file clsserverconn.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.9.3.10 void client::clsServerConn::handleOwnUserChange ( jsonCommand & *comm* ) [private]

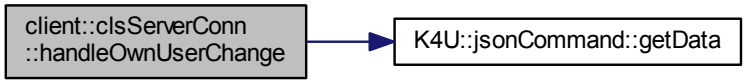
Another client has changed our own user status.

**Parameters**

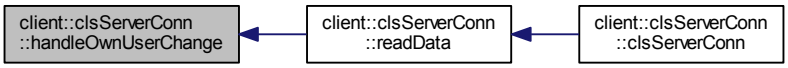
| | |
|---|---|
| *comm* | The jsonCommand containing all the data |

Definition at line 69 of file clsserverconn.cpp.

Here is the call graph for this function:



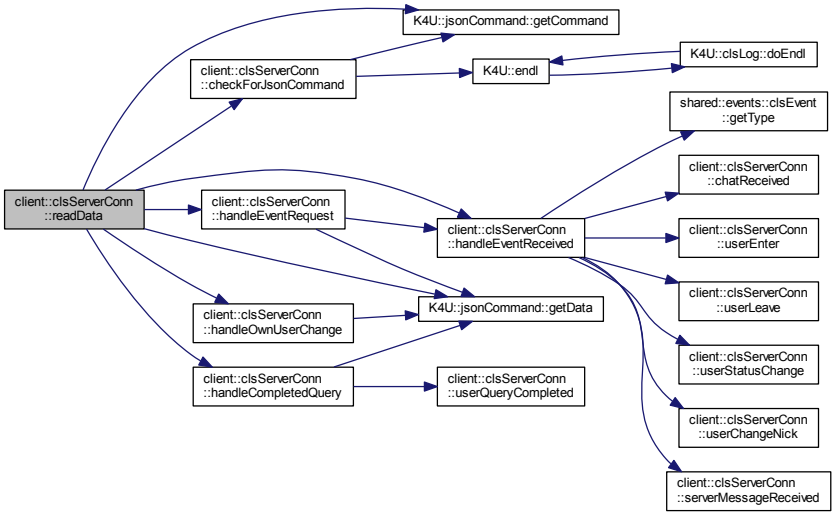

Here is the caller graph for this function:




#### 7.9.3.11 void client::clsServerConn::readData ( ) [slot]

Definition at line 139 of file clsserverconn.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

#### 7.9.3.12 void client::clsServerConn::sendCommand ( jsonCommand & *toSend* )

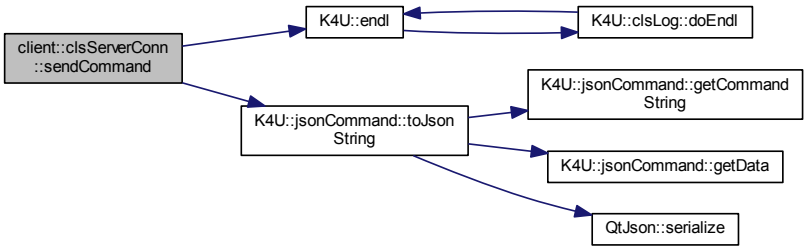
Send a JSON command to the server.

**Parameters**

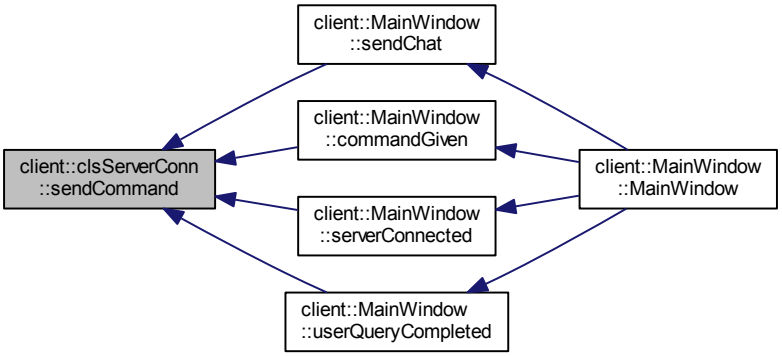
| | |
|---|---|
| *toSend* | jsonCommand that should be sent |

Definition at line 38 of file clsserverconn.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:



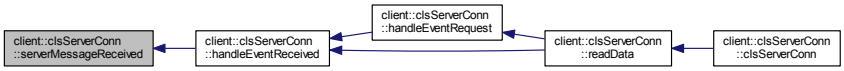
#### 7.9.3.13 void client::clsServerConn::serverMessageReceived ( eventServerMessage * *theMessage* ) [signal]

Triggered when the server wants us to know something.

**Parameters**

| | |
|---|---|
| *theMessage* | The message it sends |

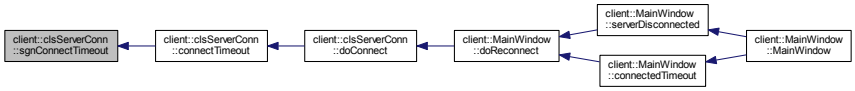
Here is the caller graph for this function:




#### 7.9.3.14 void client::clsServerConn::sgnConnectTimeout ( ) [signal]

Triggered when it cannot connect.

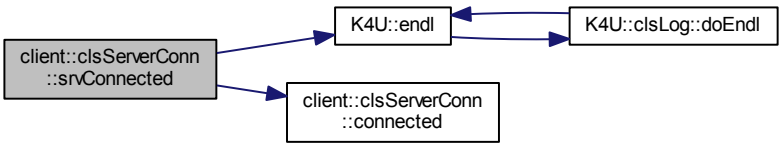
Here is the caller graph for this function:




#### 7.9.3.15 void client::clsServerConn::srvConnected ( ) [slot]

Definition at line 46 of file clsserverconn.cpp.

Here is the call graph for this function:

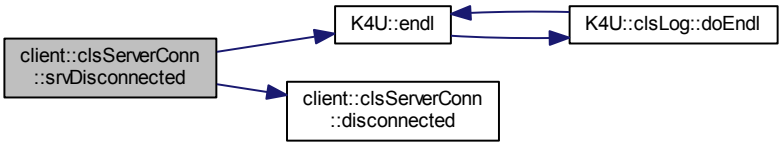


Here is the caller graph for this function:

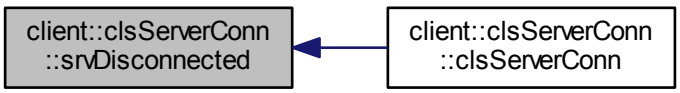
#### 7.9.3.16 void client::clsServerConn::srvDisconnected ( ) [slot]

Definition at line 54 of file clsserverconn.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.9.3.17 void client::clsServerConn::userChangeNick ( eventUserChangeNick * *changeEvent* ) [signal]

Emitted when a user has changed his nick (nick|status)

**Parameters**

| | |
|---|---|
| *changeEvent* | And eventUserChangeNick containing more data |

**Note**

Not to be confused with userStatusChange

Here is the caller graph for this function:

#### 7.9.3.18 void client::clsServerConn::userEnter ( eventUserJoin * *newUser* ) [signal]

Triggered when a user enters the channel.

**Parameters**

| | |
|---|---|
| *newUser* | An eventUserJoin containing more data |

Here is the caller graph for this function:

#### 7.9.3.19 void client::clsServerConn::userLeave ( eventUserLeave * *leaveEvent* ) [signal]

Triggered when a user left the channel.

**Parameters**

| | |
|---|---|
| *leaveEvent* | An eventUserLeave containing more data |

Here is the caller graph for this function:

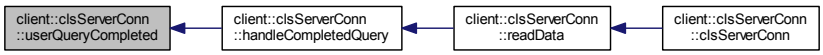
#### 7.9.3.20 void client::clsServerConn::userQueryCompleted ( QVector< ircUser * > *users* ) [signal]

Emitted when the server has sent us all the users.

**Parameters**

| | |
|---|---|
| *users* | A QVector<ircUser*> of all the users |

Here is the caller graph for this function:




#### 7.9.3.21 void client::clsServerConn::userStatusChange ( eventUserChangeStatus * *changeEvent* ) [signal]

Emitted when a user has changed his status (nick|status)

**Parameters**

| | |
|---|---|
| *changeEvent* | An eventUserChangeStatus containing more data |

**Note**

Not to be confused with userChangeNick!

Here is the caller graph for this function:

### 7.9.4 Member Data Documentation

#### 7.9.4.1 QString* client::clsServerConn::buffer [private]

Definition at line 158 of file clsserverconn.h.

#### 7.9.4.2 QTimer* client::clsServerConn::connectTimer [private]

Definition at line 163 of file clsserverconn.h.

#### 7.9.4.3 int client::clsServerConn::depth [private]

Definition at line 159 of file clsserverconn.h.

#### 7.9.4.4 QString client::clsServerConn::hostName [private]

Definition at line 154 of file clsserverconn.h.

#### 7.9.4.5 bool client::clsServerConn::isConnected [private]

Definition at line 164 of file clsserverconn.h.

#### 7.9.4.6 bool client::clsServerConn::isConnecting [private]

Definition at line 165 of file clsserverconn.h.

#### 7.9.4.7 clsLog client::clsServerConn::log [private]

Definition at line 161 of file clsserverconn.h.

#### 7.9.4.8 int client::clsServerConn::port [private]

Definition at line 155 of file clsserverconn.h.

#### 7.9.4.9 QTcpSocket* client::clsServerConn::sock [private]

Definition at line 157 of file clsserverconn.h.

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

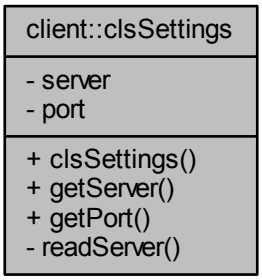
- C++/customIrcClient/clsserverconn.h
- C++/customIrcClient/clsserverconn.cpp

## 7.10 client::clsSettings Class Reference

Settings class containing all the variables needed for the client.

```
#include <clssettings.h>
```

Collaboration diagram for client::clsSettings:




### Public Member Functions

- clsSettings (const QString &filePath)

  *clsSettings Constructor*
- QString getServer () const

  *Gets the hostname of where the server is running.*
- int getPort () const

  *Gets the port on which the server is listening.*

### Private Member Functions

- void readServer (QXmlStreamReader &reader)

### Private Attributes

- QString server
- int port

## 7.10.1 Detailed Description

Settings class containing all the variables needed for the client.

**Author**

Koen Beckers (K-4U) Gets the settings from the xml file called settings.xml

Definition at line 16 of file clssettings.h.

## 7.10.2 Constructor & Destructor Documentation

### 7.10.2.1 client::clsSettings::clsSettings ( const QString & *filePath* )

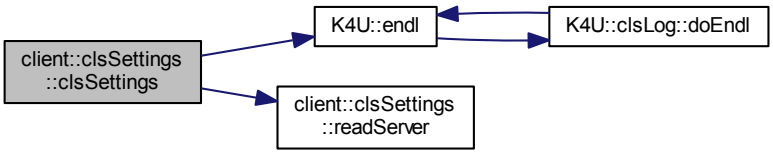
clsSettings Constructor

**Parameters**

| | |
|---|---|
| *filePath* | the path where the settings.xml file is located |

Definition at line 5 of file clssettings.cpp.

Here is the call graph for this function:




## 7.10.3 Member Function Documentation

### 7.10.3.1 int client::clsSettings::getPort ( ) const

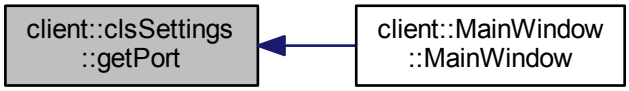
Gets the port on which the server is listening.

**Returns**

The port on which the server is listening

Definition at line 36 of file clssettings.cpp.

Here is the caller graph for this function:




#### 7.10.3.2 QString client::clsSettings::getServer ( ) const

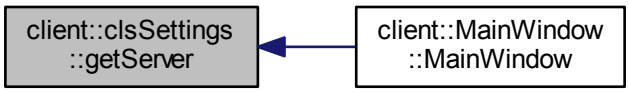
Gets the hostname of where the server is running.

**Returns**

The hostname of where the server is running

Definition at line 32 of file clssettings.cpp.

Here is the caller graph for this function:




#### 7.10.3.3 void client::clsSettings::readServer ( QXmlStreamReader & *reader* ) [private]

Definition at line 41 of file clssettings.cpp.

Here is the caller graph for this function:

### 7.10.4 Member Data Documentation

#### 7.10.4.1 int client::clsSettings::port [private]

Definition at line 42 of file clssettings.h.

#### 7.10.4.2 QString client::clsSettings::server [private]

Definition at line 41 of file clssettings.h.

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

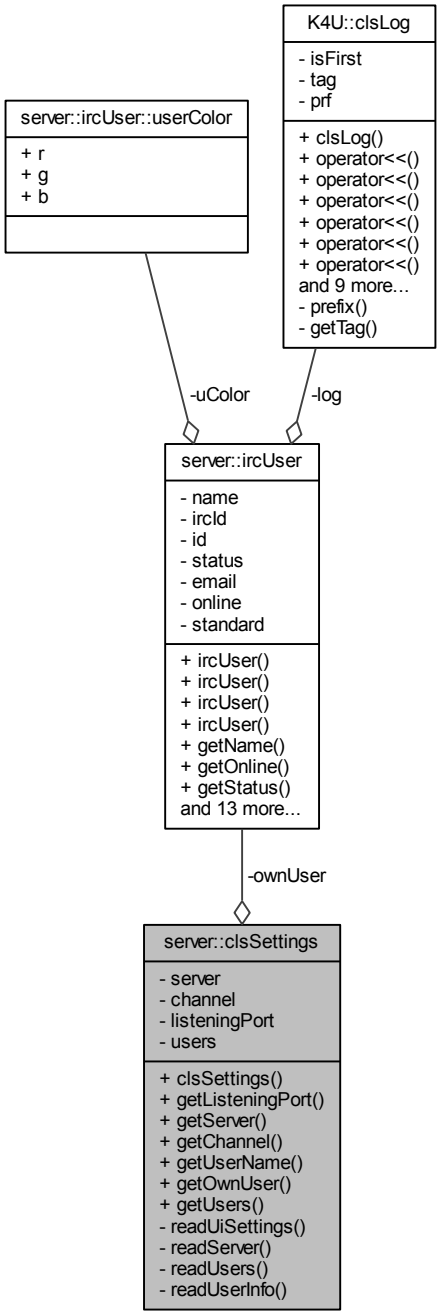
- C++/customIrcClient/clssettings.h
- C++/customIrcClient/clssettings.cpp

## 7.11 server::clsSettings Class Reference

Settings class.

```
#include <clssettings.h>
```

Collaboration diagram for server::clsSettings:




## Public Member Functions

- clsSettings (const QString &filePath)

  *clsSettings Constructor*
- int getListeningPort () const

  *getListeningPort*
- QString getServer () const

*getServer*

- QString getChannel () const

  *getChannel*
- QString getUserName () const

  *getUserName*
- ircUser * getOwnUser () const

  *getOwnUser*
- QVector< ircUser * > getUsers () const

  *getUsers*

### Private Member Functions

- void readUiSettings (QXmlStreamReader &reader)

  *readUiSettings*
- void readServer (QXmlStreamReader &reader)

  *readServer*
- void readUsers (QXmlStreamReader &reader)

  *readUsers*
- void readUserInfo (QXmlStreamReader &reader)

  *readUserInfo*

### Private Attributes

- QString server

  *server Stores the irc server host*
- QString channel

  *channel Stores the irc channel*
- ircUser * ownUser

  *ownUser Stores the own user to connect with*
- int listeningPort

  *listeningPort The port for the uiServer to listen on*
- QVector< ircUser * > users

  *users The 'standard' users*

## 7.11.1 Detailed Description

Settings class.

**Author**

Koen Beckers (K-4U) Gets the settings from the xml file called settings.xml

Definition at line 19 of file clssettings.h.

## 7.11.2 Constructor & Destructor Documentation

### 7.11.2.1 server::clsSettings::clsSettings ( const QString & *filePath* )

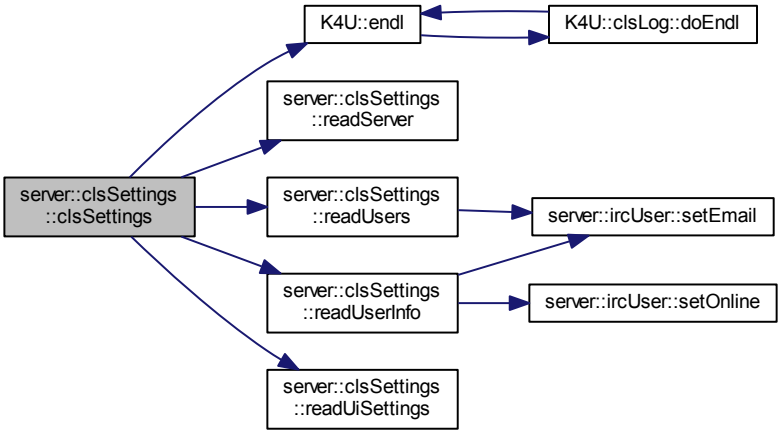
clsSettings Constructor

**Parameters**

| | |
|---|---|
| *filePath* | the path where the settings.xml file is located |

Definition at line 4 of file clssettings.cpp.

Here is the call graph for this function:




### 7.11.3 Member Function Documentation

#### 7.11.3.1 QString server::clsSettings::getChannel ( ) const

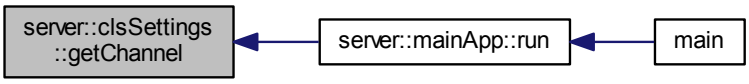
getChannel

**Returns**

the channel to connect to.

Definition at line 50 of file clssettings.cpp.

Here is the caller graph for this function:




#### 7.11.3.2 int server::clsSettings::getListeningPort ( ) const

getListeningPort

**Returns**

the port where the server should listen on

Definition at line 42 of file clssettings.cpp.

Here is the caller graph for this function:

#### 7.11.3.3 ircUser * server::clsSettings::getOwnUser ( ) const

getOwnUser

**Returns**

our own user as an ircUser object.

Definition at line 58 of file clssettings.cpp.

Here is the caller graph for this function:

#### 7.11.3.4 QString server::clsSettings::getServer ( ) const

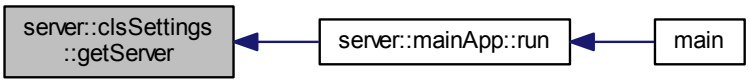
getServer

**Returns**

the host where the IRC Server is running

Definition at line 46 of file clssettings.cpp.

Here is the caller graph for this function:




#### 7.11.3.5 QString server::clsSettings::getUserName ( ) const

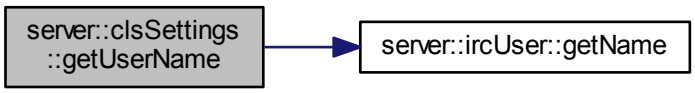
getUserName

**Returns**

the username to use for the bot.

Definition at line 54 of file clssettings.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.11.3.6 QVector< ircUser * > server::clsSettings::getUsers ( ) const

getUsers

**Returns**

the 'standard' users which are always online.

Definition at line 62 of file clssettings.cpp.

Here is the caller graph for this function:

#### 7.11.3.7 void server::clsSettings::readServer ( QXmlStreamReader & *reader* ) [private]

readServer

**Parameters**

| | |
|---|---|
| *reader* | Parses the xml file opened by reader and fetches irc server info |

Definition at line 77 of file clssettings.cpp.

Here is the caller graph for this function:

#### 7.11.3.8 void server::clsSettings::readUiSettings ( QXmlStreamReader & *reader* ) [private]

readUiSettings

**Parameters**

| | |
|---|---|
| *reader* | Parses the xml file opened by reader and fetches uiSettings |

Definition at line 66 of file clssettings.cpp.

Here is the caller graph for this function:

#### 7.11.3.9 void server::clsSettings::readUserInfo ( QXmlStreamReader & *reader* ) [private]

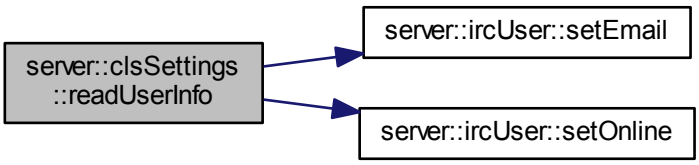
readUserInfo

**Parameters**

| | |
|---|---|
| *reader* | Parses the xml file opened by reader and fetches the own user settings |

Definition at line 115 of file clssettings.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.11.3.10 void server::clsSettings::readUsers ( QXmlStreamReader & *reader* ) [private]

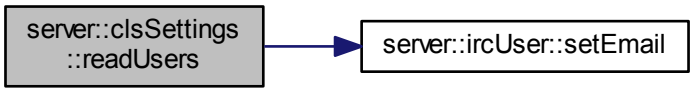
readUsers

**Parameters**

| | |
|---|---|
| *reader* | Parses the xml file opened by reader and fetches the standard users |

Definition at line 89 of file clssettings.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

### 7.11.4 Member Data Documentation

#### 7.11.4.1 QString server::clsSettings::channel [private]

channel Stores the irc channel

Definition at line 104 of file clssettings.h.

#### 7.11.4.2 int server::clsSettings::listeningPort [private]

listeningPort The port for the uiServer to listen on

Definition at line 116 of file clssettings.h.

#### 7.11.4.3 ircUser* server::clsSettings::ownUser [private]

ownUser Stores the own user to connect with

Definition at line 110 of file clssettings.h.

#### 7.11.4.4 QString server::clsSettings::server [private]

server Stores the irc server host

Definition at line 98 of file clssettings.h.

#### 7.11.4.5 QVector<ircUser*> server::clsSettings::users [private]

users The 'standard' users

Definition at line 122 of file clssettings.h.

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

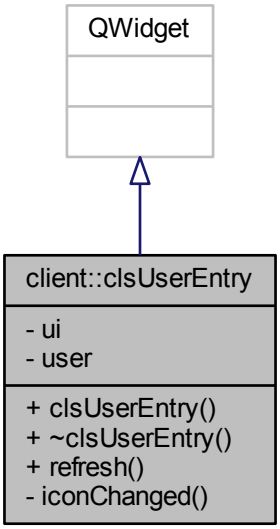
- C++/customIrcServer/clssettings.h
- C++/customIrcServer/clssettings.cpp

## 7.12 client::clsUserEntry Class Reference

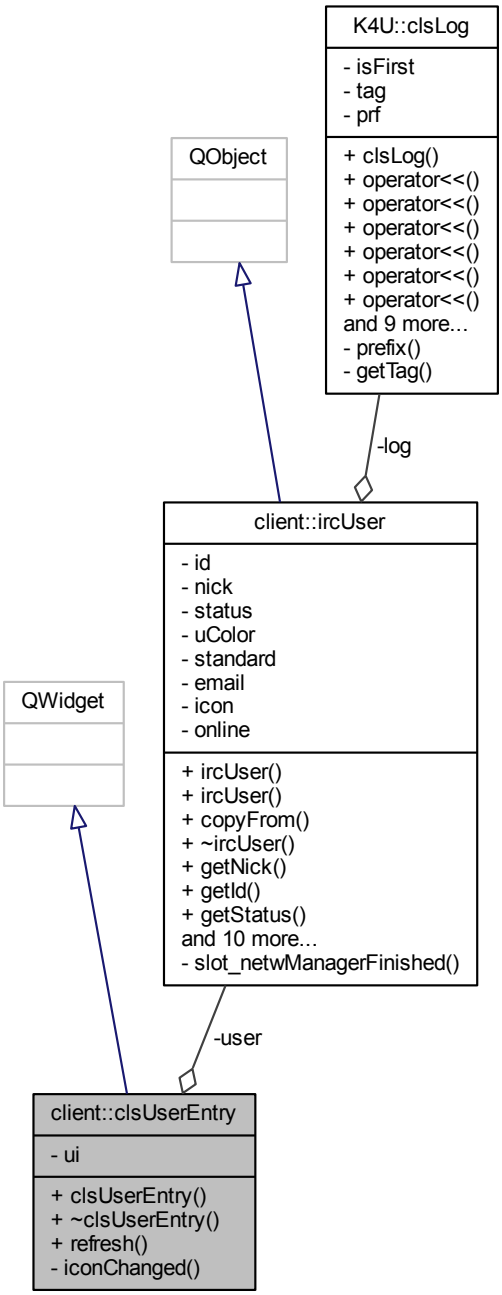
UI segment for a user entry in the user list.

`#include <clsuserentry.h>`

Inheritance diagram for client::clsUserEntry:



Collaboration diagram for client::clsUserEntry:




## Public Member Functions

- clsUserEntry (ircUser *user, QWidget *parent=0)
- ~clsUserEntry ()
- void refresh ()

### Private Slots

- void iconChanged ()

### Private Attributes

- Ui::clsUserEntry * ui
- ircUser * user

## 7.12.1 Detailed Description

UI segment for a user entry in the user list.

**Author**

Koen Beckers (K-4U)

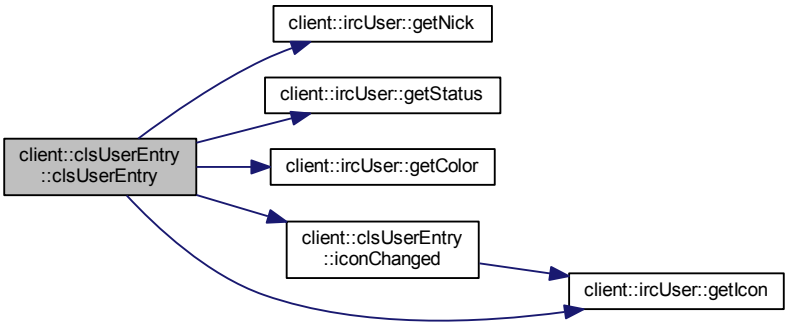
Definition at line 17 of file clsuserentry.h.

## 7.12.2 Constructor & Destructor Documentation

### 7.12.2.1 client::clsUserEntry::clsUserEntry ( ircUser * *user*, QWidget * *parent* = 0 ) [explicit]

Definition at line 6 of file clsuserentry.cpp.

Here is the call graph for this function:




### 7.12.2.2 client::clsUserEntry::~clsUserEntry ( )

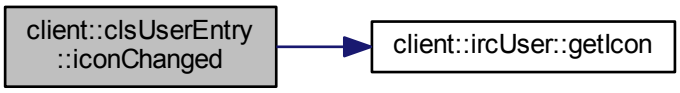
Definition at line 29 of file clsuserentry.cpp.

## 7.12.3 Member Function Documentation

### 7.12.3.1 void client::clsUserEntry::iconChanged ( ) [private], [slot]

Definition at line 50 of file clsuserentry.cpp.

Here is the call graph for this function:

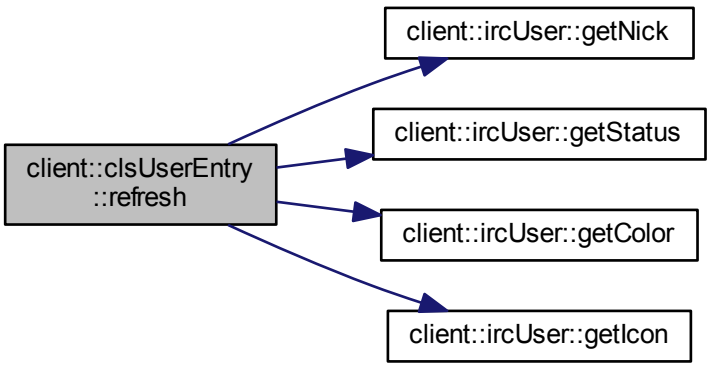



Here is the caller graph for this function:

#### 7.12.3.2 void client::clsUserEntry::refresh ( )

Definition at line 33 of file clsuserentry.cpp.

Here is the call graph for this function:




### 7.12.4 Member Data Documentation

#### 7.12.4.1 Ui::clsUserEntry* client::clsUserEntry::ui [private]

Definition at line 30 of file clsuserentry.h.

#### 7.12.4.2 ircUser* client::clsUserEntry::user [private]

Definition at line 32 of file clsuserentry.h.

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

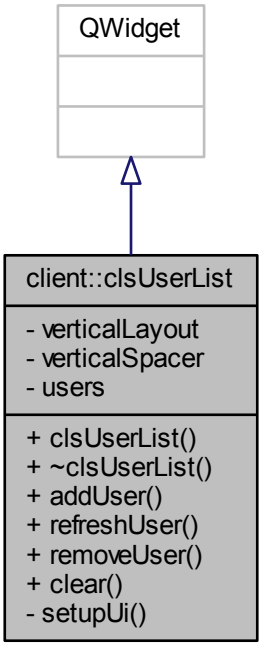
- C++/customIrcClient/desktop/clsuserentry.h
- C++/customIrcClient/desktop/clsuserentry.cpp

## 7.13 client::clsUserList Class Reference

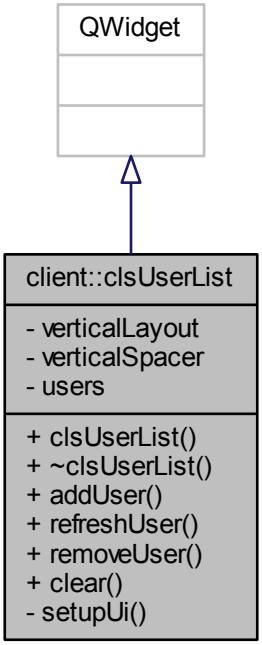
UI Segment for user list.

`#include <clsuserlist.h>`

Inheritance diagram for client::clsUserList:



Collaboration diagram for client::clsUserList:




### Public Member Functions

- clsUserList (QWidget *parent=0)
- ~clsUserList ()
- void addUser (ircUser *user)

  *Adds a user to the list.*
- void refreshUser (ircUser *user)

  *Refreshes a user entry.*
- void removeUser (ircUser *user)

  *Removes a user entry.*
- void clear ()

  *Clears the list.*

### Private Member Functions

- void setupUi ()

### Private Attributes

- QVBoxLayout * verticalLayout
- QSpacerItem * verticalSpacer
- QMap< ircUser *, clsUserEntry * > users

### 7.13.1 Detailed Description

UI Segment for user list.

**Author**

Koen Beckers (K-4U)

Definition at line 21 of file clsuserlist.h.

### 7.13.2 Constructor & Destructor Documentation

#### 7.13.2.1 client::clsUserList::clsUserList ( QWidget * *parent* = 0 ) [explicit]

Definition at line 5 of file clsuserlist.cpp.

Here is the call graph for this function:

#### 7.13.2.2 client::clsUserList::~clsUserList ( )

Definition at line 11 of file clsuserlist.cpp.

### 7.13.3 Member Function Documentation

#### 7.13.3.1 void client::clsUserList::addUser ( ircUser * *user* )

Adds a user to the list.

**Parameters**

| | |
|---|---|
| *user* | The user to be added |

Definition at line 15 of file clsuserlist.cpp.

#### 7.13.3.2 void client::clsUserList::clear ( )

Clears the list.

Definition at line 39 of file clsuserlist.cpp.

Here is the call graph for this function:

#### 7.13.3.3 void client::clsUserList::refreshUser ( ircUser * *user* )

Refreshes a user entry.

**Parameters**

| | |
|---|---|
| *user* | The user to be refreshed |

Definition at line 28 of file clsuserlist.cpp.

#### 7.13.3.4 void client::clsUserList::removeUser ( ircUser * *user* )

Removes a user entry.

**Parameters**

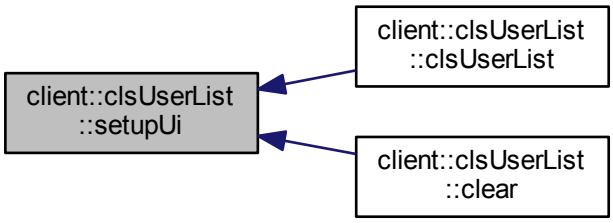
| | |
|---|---|
| *user* | The user to be removed |

Definition at line 32 of file clsuserlist.cpp.

#### 7.13.3.5 void client::clsUserList::setupUi ( ) [private]

Definition at line 43 of file clsuserlist.cpp.

Here is the caller graph for this function:



### 7.13.4 Member Data Documentation

#### 7.13.4.1 QMap<ircUser*,clsUserEntry*> client::clsUserList::users [private]

Definition at line 54 of file clsuserlist.h.

#### 7.13.4.2 QVBoxLayout* client::clsUserList::verticalLayout [private]

Definition at line 51 of file clsuserlist.h.

#### 7.13.4.3 QSpacerItem* client::clsUserList::verticalSpacer [private]

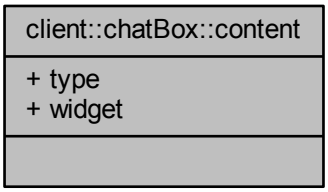
Definition at line 52 of file clsuserlist.h.

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

- C++/customIrcClient/desktop/clsuserlist.h
- C++/customIrcClient/desktop/clsuserlist.cpp

## 7.14 client::chatBox::content Struct Reference

Collaboration diagram for client::chatBox::content:




### Public Attributes

- widgetTypes type
- QWidget * widget

### 7.14.1 Detailed Description

Definition at line 30 of file chatbox.h.

### 7.14.2 Member Data Documentation

#### 7.14.2.1 widgetTypes client::chatBox::content::type

Definition at line 31 of file chatbox.h.

#### 7.14.2.2 QWidget* client::chatBox::content::widget

Definition at line 32 of file chatbox.h.

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

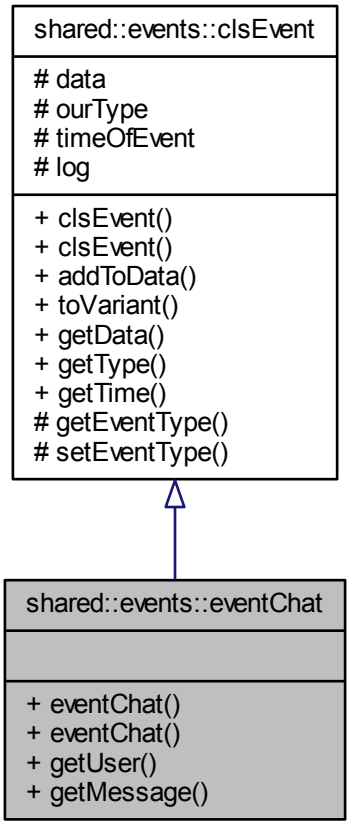
- C++/customIrcClient/chatbox.h

## 7.15 shared::events::eventChat Class Reference

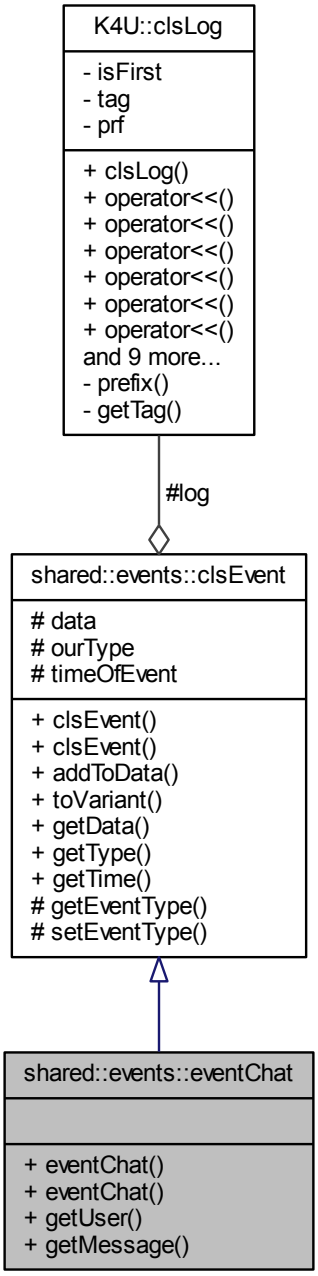
Class containing a chatEvent.

```
#include <eventchat.h>
```

Inheritance diagram for shared::events::eventChat:



Collaboration diagram for shared::events::eventChat:




## Public Member Functions

- eventChat (int userId, QString message)

  *eventChat constructor taking a userId and a message*
- eventChat (QVariantMap data)

  *Initializer function that accepts data immediately.*
- int getUser () const

*Gets the DB-ID of the user that sent the message.*

- QString getMessage () const

  *Gets the chat message.*

## Additional Inherited Members

### 7.15.1 Detailed Description

Class containing a chatEvent.

**Author**

Koen Beckers (K-4U)

Definition at line 11 of file eventchat.h.

### 7.15.2 Constructor & Destructor Documentation

#### 7.15.2.1 shared::events::eventChat::eventChat ( int *userId*, QString *message* )

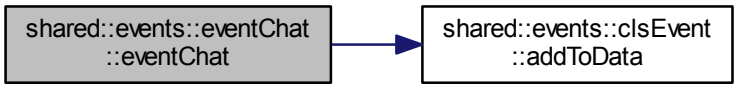
eventChat constructor taking a userId and a message

**Parameters**

| | |
|---|---|
| *userId* | The DB-ID of the user that sent the message |
| *message* | The message that was sent |

Definition at line 6 of file eventchat.cpp.

Here is the call graph for this function:




#### 7.15.2.2 shared::events::eventChat::eventChat ( QVariantMap *data* )

Initializer function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 12 of file eventchat.cpp.

### 7.15.3 Member Function Documentation

#### 7.15.3.1 QString shared::events::eventChat::getMessage ( ) const

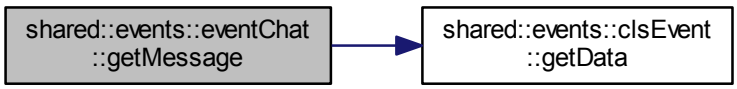
Gets the chat message.

**Returns**

The chat message

Definition at line 20 of file eventchat.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.15.3.2 int shared::events::eventChat::getUser ( ) const

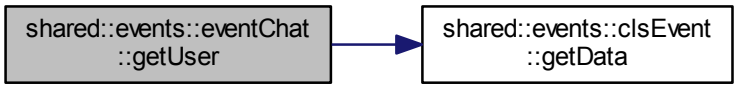
Gets the DB-ID of the user that sent the message.

**Returns**

The DB-ID of the user that sent the message

Definition at line 16 of file eventchat.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

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

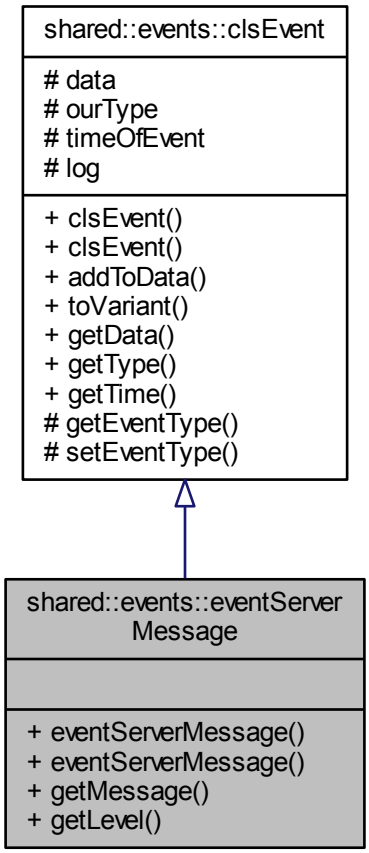
- C++/resources/events/eventchat.h
- C++/resources/events/eventchat.cpp

## 7.16 shared::events::eventServerMessage Class Reference

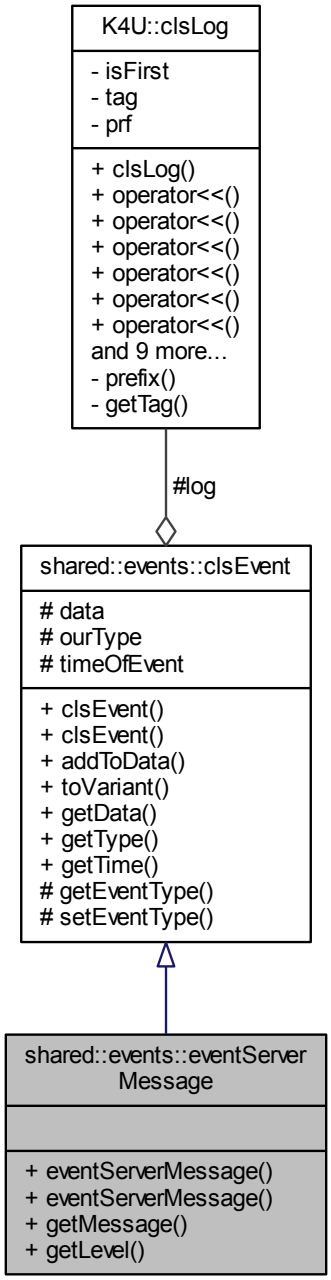
Class that contains data for when the server wants to say something.

`#include <eventservermessage.h>`

Inheritance diagram for shared::events::eventServerMessage:



Collaboration diagram for shared::events::eventServerMessage:




## Public Member Functions

- eventServerMessage (QString message, level lvl)

  *eventServerMessage constructor taking a message*
- eventServerMessage (QVariantMap data)

  *Initializer function that accepts data immediately.*
- QString getMessage () const

*Gets the message.*

- level getLevel () const

  *Gets the level of the message.*

## Additional Inherited Members

### 7.16.1 Detailed Description

Class that contains data for when the server wants to say something.

**Author**

Koen Beckers (K-4U)

Definition at line 15 of file eventservermessage.h.

### 7.16.2 Constructor & Destructor Documentation

#### 7.16.2.1 shared::events::eventServerMessage::eventServerMessage ( QString *message*, level *lvl* )

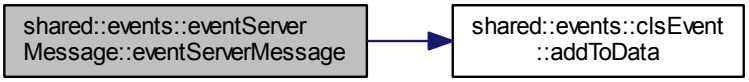
eventServerMessage constructor taking a message

**Parameters**

| | |
|---|---|
| *message* | The message that was sent |

Definition at line 6 of file eventservermessage.cpp.

Here is the call graph for this function:




#### 7.16.2.2 shared::events::eventServerMessage::eventServerMessage ( QVariantMap *data* )

Initializer function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 12 of file eventservermessage.cpp.

### 7.16.3 Member Function Documentation

#### 7.16.3.1 level shared::events::eventServerMessage::getLevel ( ) const

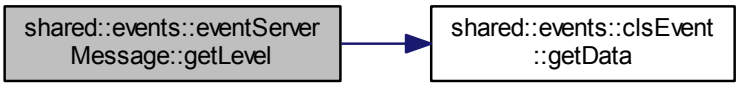
Gets the level of the message.

**Returns**

The level of the message

Definition at line 20 of file eventservermessage.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.16.3.2 QString shared::events::eventServerMessage::getMessage ( ) const

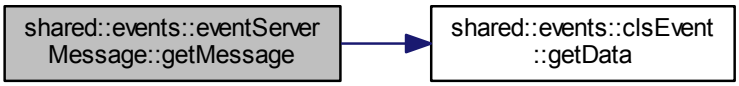
Gets the message.

**Returns**

The message

Definition at line 16 of file eventservermessage.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

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

- C++/resources/events/eventservermessage.h

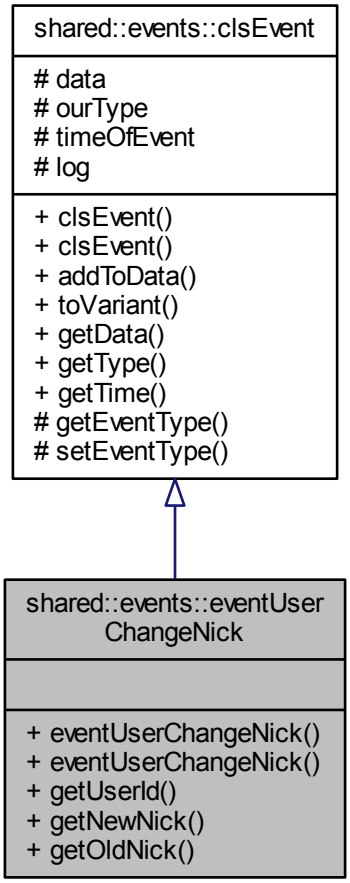
- C++/resources/events/eventservermessage.cpp

## 7.17 shared::events::eventUserChangeNick Class Reference

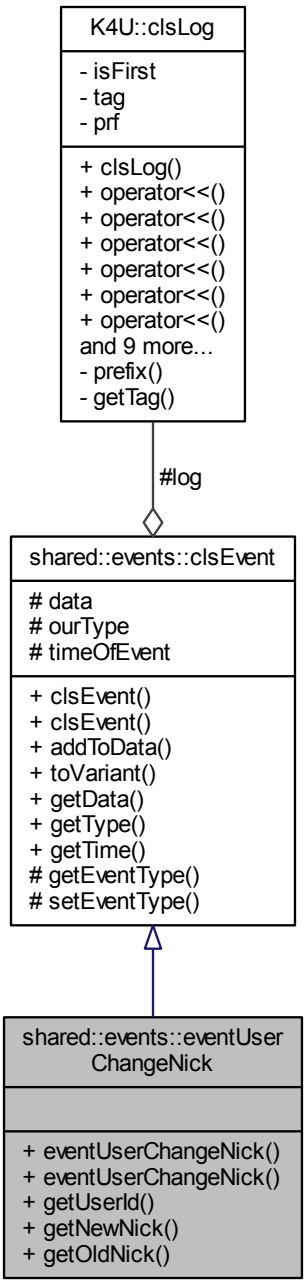
Class containing a user who changes his nick-event (nick|status)

```
#include <eventuserchangenick.h>
```

Inheritance diagram for shared::events::eventUserChangeNick:



Collaboration diagram for shared::events::eventUserChangeNick:




## Public Member Functions

- eventUserChangeNick (const int userId, const QString newNick, const QString oldNick)

  *eventUserChangeNick constructor taking only a userId*
- eventUserChangeNick (QVariantMap data)

  *Constructor function that accepts data immediately.*
- int getUserId () const

*Gets the DB-ID of the user that changed nick.*

- QString getNewNick () const

  *Gets the new nick of the user.*

- QString getOldNick () const

  *Gets the old nick of the user.*

## Additional Inherited Members

### 7.17.1 Detailed Description

Class containing a user who changes his nick-event (nick|status)

**Author**

Koen Beckers (K-4U)

Definition at line 11 of file eventuserchangenick.h.

### 7.17.2 Constructor & Destructor Documentation

#### 7.17.2.1 shared::events::eventUserChangeNick::eventUserChangeNick ( const int *userId*, const QString *newNick*, const QString *oldNick* )

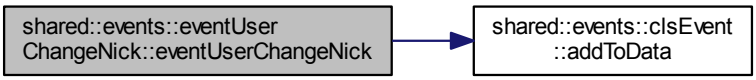
eventUserChangeNick constructor taking only a userId

**Parameters**

| | |
|---|---|
| *userId* | The DB-ID of the user that changed nick |
| *newNick* | The new nickname of the user |
| *oldNick* | The old nickname of the user |

Definition at line 6 of file eventuserchangenick.cpp.

Here is the call graph for this function:




#### 7.17.2.2 shared::events::eventUserChangeNick::eventUserChangeNick ( QVariantMap *data* )

Constructor function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 15 of file eventuserchangenick.cpp.

### 7.17.3 Member Function Documentation

#### 7.17.3.1 QString shared::events::eventUserChangeNick::getNewNick ( ) const

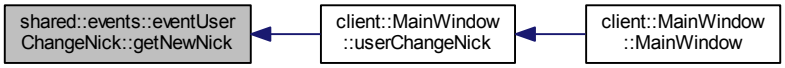
Gets the new nick of the user.

**Returns**

The new nick of the user

Definition at line 23 of file eventuserchangenick.cpp.

Here is the caller graph for this function:




#### 7.17.3.2 QString shared::events::eventUserChangeNick::getOldNick ( ) const

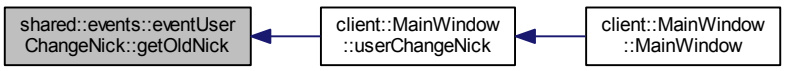
Gets the old nick of the user.

**Returns**

The old nick of the user

Definition at line 27 of file eventuserchangenick.cpp.

Here is the caller graph for this function:




#### 7.17.3.3 int shared::events::eventUserChangeNick::getUserId ( ) const

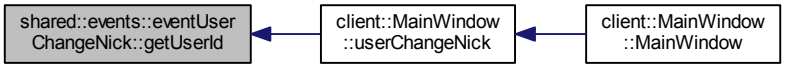
Gets the DB-ID of the user that changed nick.

**Returns**

The DB-ID of the user that changed status

Definition at line 19 of file eventuserchangenick.cpp.

Here is the caller graph for this function:




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

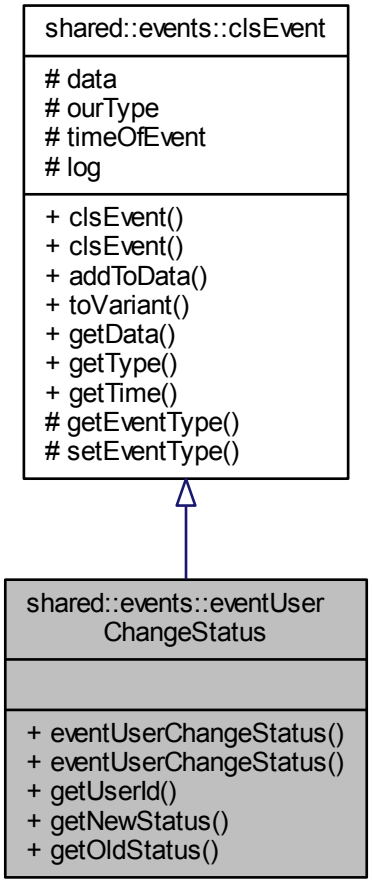
- C++/resources/events/eventuserchangenick.h
- C++/resources/events/eventuserchangenick.cpp

## 7.18 shared::events::eventUserChangeStatus Class Reference

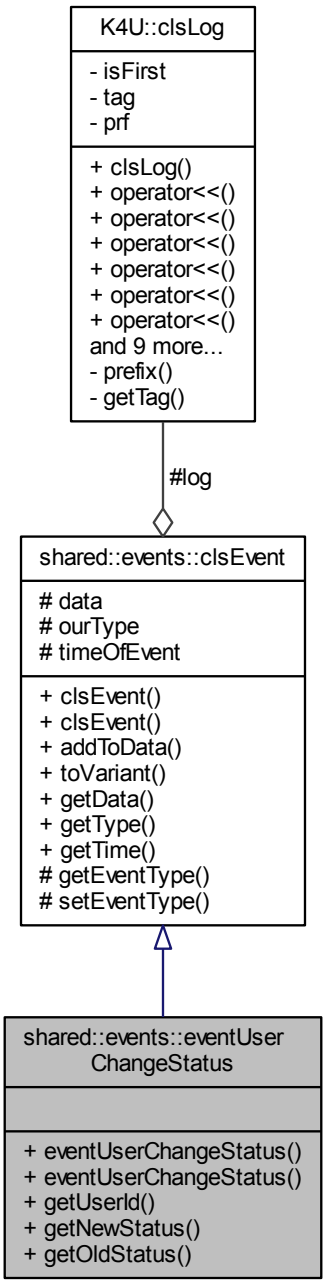
Class containing a user who changes his status-event (nick|status)

```
#include <eventuserchangestatus.h>
```

Inheritance diagram for shared::events::eventUserChangeStatus:



Collaboration diagram for shared::events::eventUserChangeStatus:




## Public Member Functions

- eventUserChangeStatus (const int userId, const QString newStatus, const QString oldStatus)

  *eventUserChangeStatus constructor taking only a userId*
- eventUserChangeStatus (QVariantMap data)

  *Constructor function that accepts data immediately.*
- int getUserId () const

*Gets the DB-ID of the user that changed status.*

- QString getNewStatus () const

  *Gets the new status of the user.*

- QString getOldStatus () const

  *Gets the old status of the user.*

## Additional Inherited Members

### 7.18.1 Detailed Description

Class containing a user who changes his status-event (nick|status)

**Author**

Koen Beckers (K-4U)

Definition at line 11 of file eventuserchangestatus.h.

### 7.18.2 Constructor & Destructor Documentation

#### 7.18.2.1 shared::events::eventUserChangeStatus::eventUserChangeStatus ( const int *userId*, const QString *newStatus*, const QString *oldStatus* )

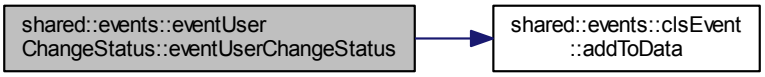
eventUserChangeStatus constructor taking only a userId

**Parameters**

| | |
|---|---|
| *userId* | The DB-ID of the user that left |
| *newStatus* | The new status of the user |
| *oldStatus* | The old status of the user |

Definition at line 6 of file eventuserchangestatus.cpp.

Here is the call graph for this function:




#### 7.18.2.2 shared::events::eventUserChangeStatus::eventUserChangeStatus ( QVariantMap *data* )

Constructor function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 15 of file eventuserchangestatus.cpp.

### 7.18.3 Member Function Documentation

#### 7.18.3.1 QString shared::events::eventUserChangeStatus::getNewStatus ( ) const

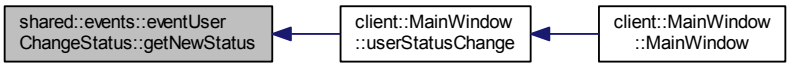
Gets the new status of the user.

**Returns**

The new status of the user

Definition at line 23 of file eventuserchangestatus.cpp.

Here is the caller graph for this function:




#### 7.18.3.2 QString shared::events::eventUserChangeStatus::getOldStatus ( ) const

Gets the old status of the user.

**Returns**

The old status of the user

Definition at line 27 of file eventuserchangestatus.cpp.

#### 7.18.3.3 int shared::events::eventUserChangeStatus::getUserId ( ) const

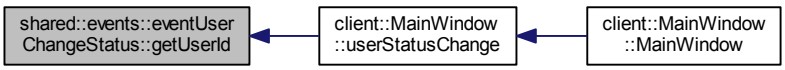
Gets the DB-ID of the user that changed status.

**Returns**

The DB-ID of the user that changed status

Definition at line 19 of file eventuserchangestatus.cpp.

Here is the caller graph for this function:




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

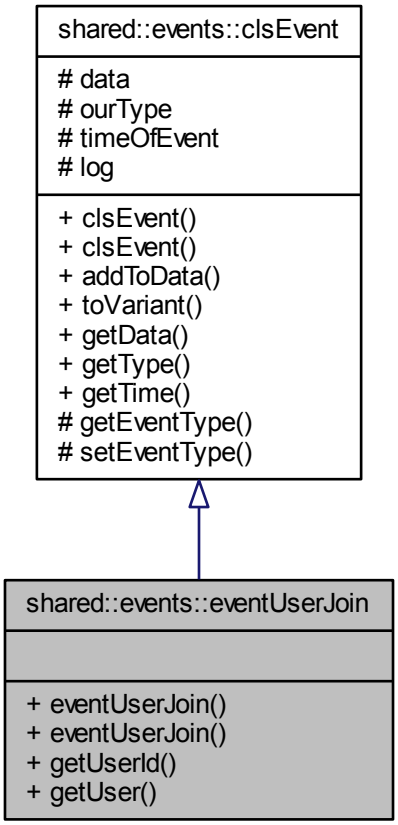
- C++/resources/events/eventuserchangestatus.h
- C++/resources/events/eventuserchangestatus.cpp

## 7.19 shared::events::eventUserJoin Class Reference

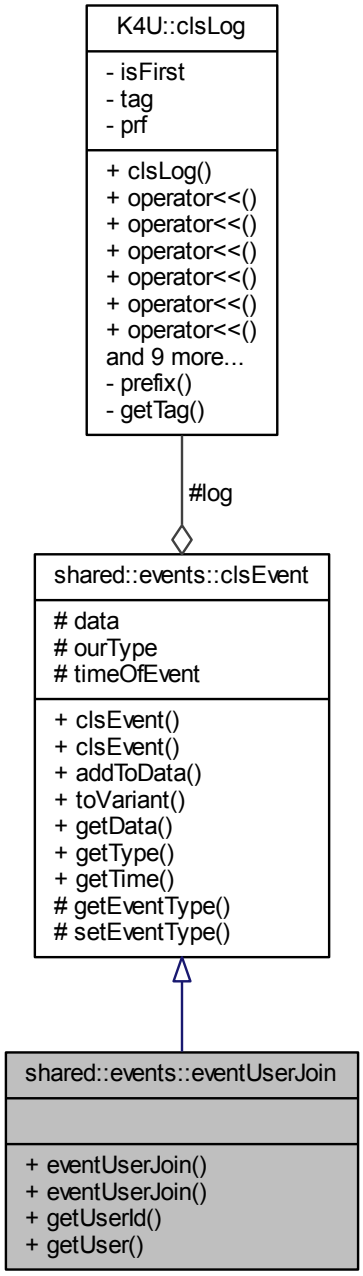
Class containing a user JOIN event.

`#include <eventuserjoin.h>`

Inheritance diagram for shared::events::eventUserJoin:



Collaboration diagram for shared::events::eventUserJoin:




## Public Member Functions

- eventUserJoin (const int userId, QVariantMap user)

  *eventUserJoin constructor taking a userId and a message*
- eventUserJoin (QVariantMap data)

  *Initializer function that accepts data immediately.*
- int getUserId () const

*Gets the DB-ID of the user that joined.*

- QVariantMap getUser () const

  *Gets a QVariantMap of the user that joined.*

## Additional Inherited Members

### 7.19.1 Detailed Description

Class containing a user JOIN event.

**Author**

Koen Beckers (K-4U)

Definition at line 11 of file eventuserjoin.h.

### 7.19.2 Constructor & Destructor Documentation

#### 7.19.2.1 shared::events::eventUserJoin::eventUserJoin ( const int *userId*, QVariantMap *user* )

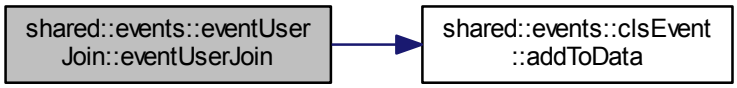
eventUserJoin constructor taking a userId and a message

**Parameters**

| | |
|---|---|
| *userId* | The DB-ID of the user that joined |
| *user* | A QVariantMap of the user that joined |

Definition at line 5 of file eventuserjoin.cpp.

Here is the call graph for this function:




#### 7.19.2.2 shared::events::eventUserJoin::eventUserJoin ( QVariantMap *data* )

Initializer function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 11 of file eventuserjoin.cpp.

### 7.19.3 Member Function Documentation

#### 7.19.3.1 QVariantMap shared::events::eventUserJoin::getUser ( ) const

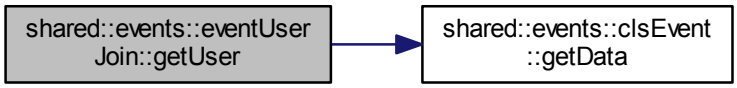
Gets a QVariantMap of the user that joined.

**Returns**

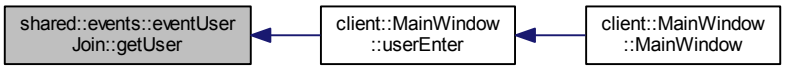
A QVariantMap of the user that joined

Definition at line 19 of file eventuserjoin.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.19.3.2 int shared::events::eventUserJoin::getUserId ( ) const

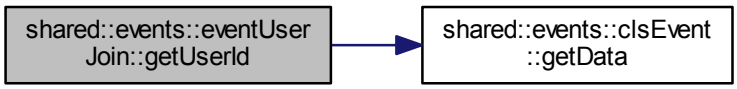
Gets the DB-ID of the user that joined.

**Returns**

The DB-ID of the user that joined

Definition at line 15 of file eventuserjoin.cpp.

Here is the call graph for this function:




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

- C++/resources/events/eventuserjoin.h

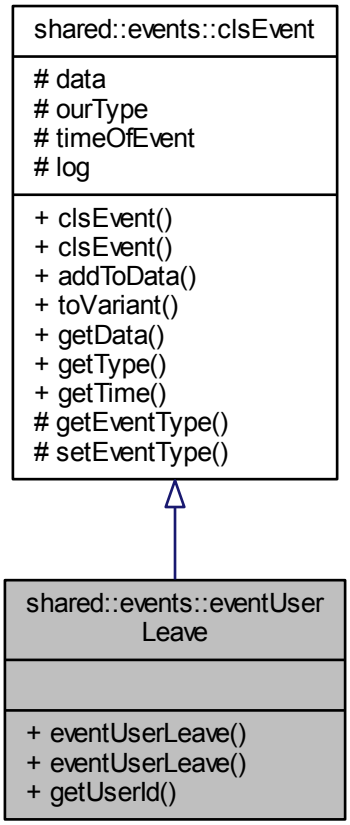
- C++/resources/events/eventuserjoin.cpp

## 7.20 shared::events::eventUserLeave Class Reference

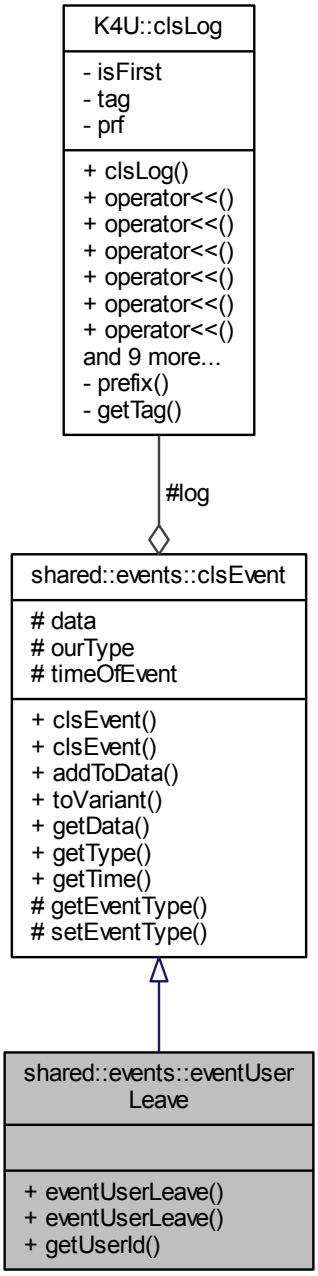
Class containing a user PART event.

```
#include <eventuserleave.h>
```

Inheritance diagram for shared::events::eventUserLeave:



Collaboration diagram for shared::events::eventUserLeave:




## Public Member Functions

- eventUserLeave (const int userId)

  *eventUserLeave constructor taking only a userId*
- eventUserLeave (QVariantMap data)

  *Initializer function that accepts data immediately.*
- int getUserId () const

*Gets the DB-ID of the user that left.*

## Additional Inherited Members

### 7.20.1 Detailed Description

Class containing a user PART event.

**Author**

Koen Beckers (K-4U)

Definition at line 11 of file eventuserleave.h.

### 7.20.2 Constructor & Destructor Documentation

#### 7.20.2.1 shared::events::eventUserLeave::eventUserLeave ( const int *userId* )

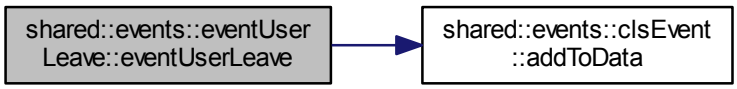
eventUserLeave constructor taking only a userId

**Parameters**

| | |
|---|---|
| *userId* | The DB-ID of the user that left |

Definition at line 6 of file eventuserleave.cpp.

Here is the call graph for this function:




#### 7.20.2.2 shared::events::eventUserLeave::eventUserLeave ( QVariantMap *data* )

Initializer function that accepts data immediately.

**Parameters**

| | |
|---|---|
| *data* | A QVariantMap containing the event data, parsed from JSON Mostly used on the receiving end. |

Definition at line 11 of file eventuserleave.cpp.

### 7.20.3 Member Function Documentation

#### 7.20.3.1 int shared::events::eventUserLeave::getUserId ( ) const

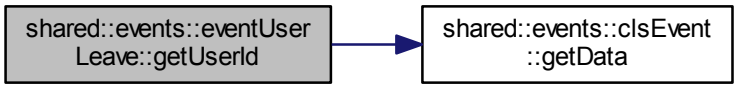
Gets the DB-ID of the user that left.

**Returns**

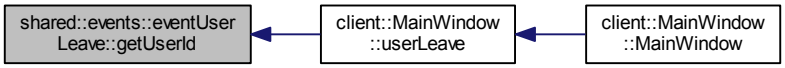
The DB-ID of the user that left

Definition at line 15 of file eventuserleave.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

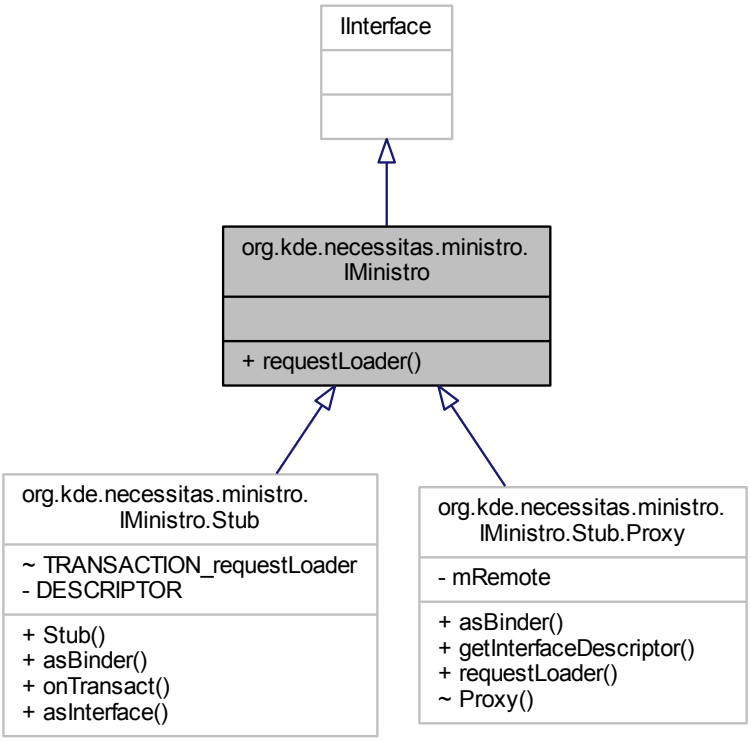



The documentation for this class was generated from the following files:

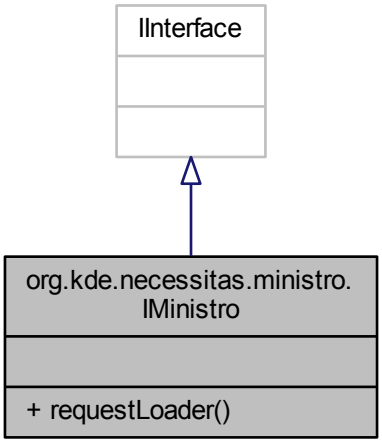
- C++/resources/events/eventuserleave.h
- C++/resources/events/eventuserleave.cpp

## 7.21 org.kde.necessitas.ministro.IMinistro Interface Reference

Inheritance diagram for org.kde.necessitas.ministro.IMinistro:




Collaboration diagram for org.kde.necessitas.ministro.IMinistro:



### Classes

- class **Stub**

### Public Member Functions

- void requestLoader (org.kde.necessitas.ministro.IMinistroCallback callback, android.os.Bundle parameters) throws android.os.RemoteException

## 7.21.1 Detailed Description

Definition at line 6 of file IMinistro.java.

## 7.21.2 Member Function Documentation

### 7.21.2.1 void org.kde.necessitas.ministro.IMinistro.requestLoader ( org.kde.necessitas.ministro.IMinistroCallback *callback,* android.os.Bundle *parameters* ) throws android.os.RemoteException

Check/download required libs to run the application

param callback - interface used by Minsitro service to notify the client when the loader is ready param parameters parameters fields:

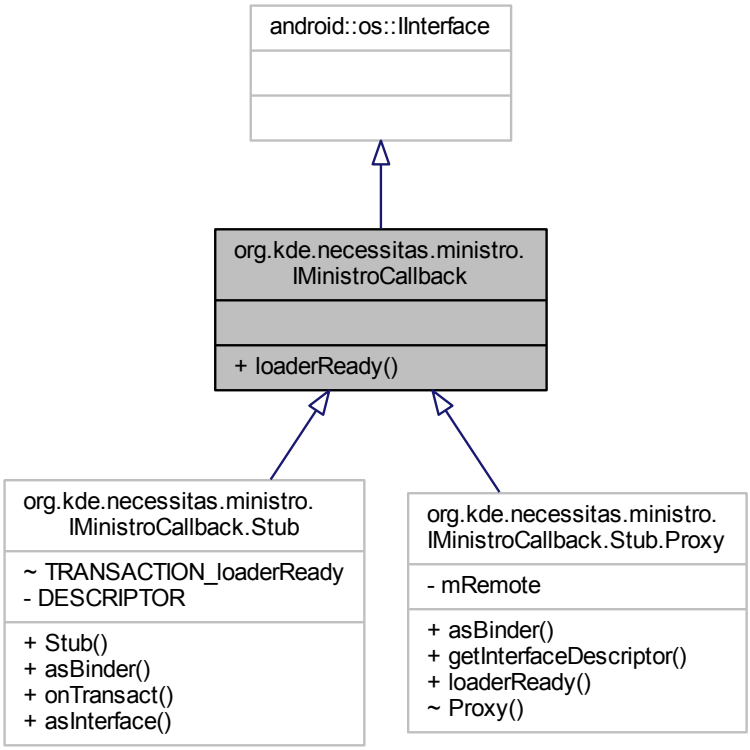
- Key Name Key type Explanations "required.modules" StringArray Required modules by your application "application.title" String Application name, used to show more informations to user "qt.provider" String Qt libs provider, currently only "necessitas" is supported. "minimum.ministro.api" Integer Minimum Ministro API level, used to check if Ministro service compatible with your application. Current API Level is 1 ! "minimum.qt.version" Integer Minimim Qt version (e.g. 0x040800, which means Qt 4.8.0, check http://doc.trolltech.com/4.8/qtglobal.html#QT_VERSION)! "3rd.party.repositories" StringArray 3rd party repositories, which should be downloaded by ministro, needs minimum.ministro.api >= 2 Check http://community.kde.org/Necessitas/Ministro for more details.

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

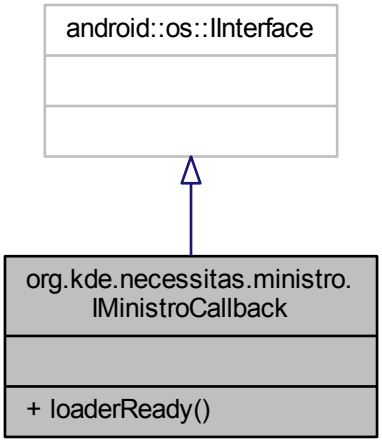
- C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java

## 7.22 org.kde.necessitas.ministro.IMinistroCallback Interface Reference

Inheritance diagram for org.kde.necessitas.ministro.IMinistroCallback:



Collaboration diagram for org.kde.necessitas.ministro.IMinistroCallback:




### Classes

- class **Stub**

### Public Member Functions

- void loaderReady (android.os.Bundle loaderParams) throws android.os.RemoteException

## 7.22.1 Detailed Description

Definition at line 6 of file IMinistroCallback.java.

## 7.22.2 Member Function Documentation

### 7.22.2.1 void org.kde.necessitas.ministro.IMinistroCallback.loaderReady ( android.os.Bundle *loaderParams* ) throws android.os.RemoteException

This method is called by the Ministro service back into the application which implements this interface.

param in - loaderParams loaderParams fields:

- Key Name Key type Explanations
- "error.code" Integer See below
- "error.message" String Missing if no error, otherwise will contain the error message translated into phone language where available.
- "dex.path" String The list of jar/apk files containing classes and resources, needed to be passed to application DexClassLoader
- "lib.path" String The list of directories containing native libraries; may be missing, needed to be passed to application DexClassLoader

- "loader.class.name" String Loader class name.

"error.code" field possible errors:

- 0 no error.
- 1 incompatible Ministro version. Ministro needs to be upgraded.
- 2 not all modules could be satisfy.
- 3 invalid parameters

This parameter will contain additional fields which are used by the loader to start your application, so it must be passed to loader.

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

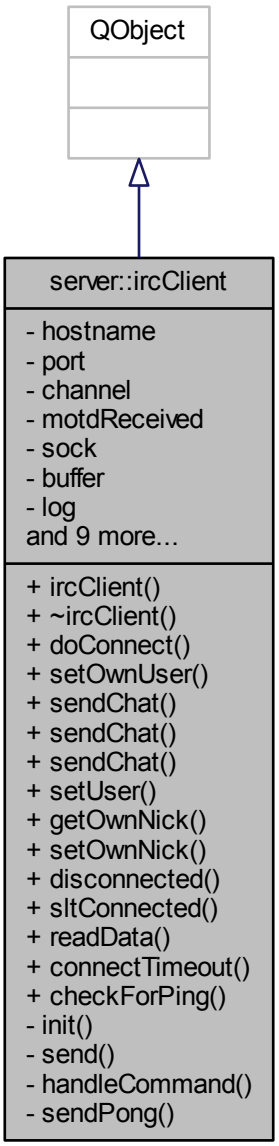
- C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java

## 7.23 server::ircClient Class Reference

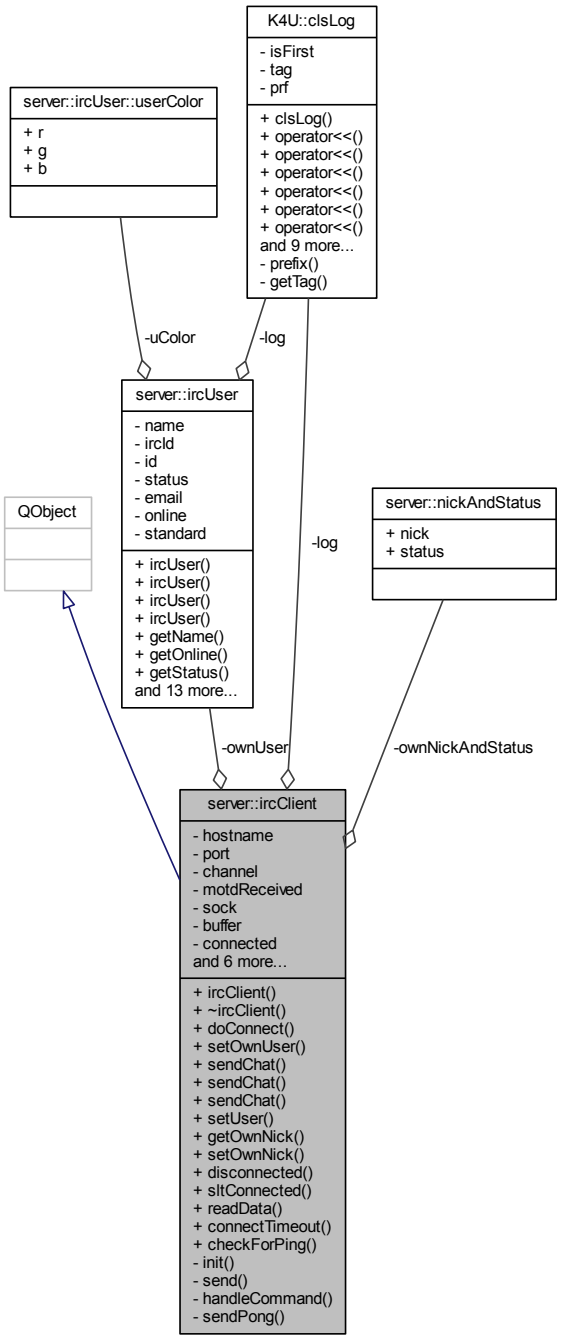
The ircClient class.

```
#include <ircclient.h>
```

Inheritance diagram for server::ircClient:



Collaboration diagram for server::ircClient:




## Public Slots

- void disconnected ()
- void sltConnected ()
- void readData ()

  *Reads data from the IRC Socket.*
- void connectTimeout ()

- void checkForPing ()

  *Runs every CHECK_PING_INTERVAL to check if the last ping was there.*

## Signals

- void startupComplete ()

  *Emitted when the client is logged in. MOTD has been received.*
- void chatReceived (const QString channel, const nickAndStatus user, const QString message)

  *Emitted when a chatmessage has been received.*
- void userOnline (const nickAndStatus nickName, const QString id, const QString channel)

  *Emitted when a user comes online with a JOIN message.*
- void userOffline (const nickAndStatus nickName, const QString id, const QString channel)

  *Emitted when a user leaves the room with a PART message.*
- void userChangeNick (const nickAndStatus oldNickName, const nickAndStatus newNickName, const QString id)

  *Emitted when a user changes his nickname (nick|Status)*
- void userChangeStatus (const nickAndStatus oldNick, const nickAndStatus newNick, const QString id)

  *Emitted when a user changes his status (nick|Status)*
- void IRCConnectTimeOut ()

  *Emitted when a connection cannot be made after MAX_CONNECT_TIMEOUT_IN_MS.*
- void IRCConnected ()

  *Emitted when the connection has been established.*
- void IRCDisconnected (QString reason)

  *Emitted when the connection was cut!*

## Public Member Functions

- ircClient (const QString &hostname, const QString &channel, const QString &nick, const int port=PORT)

  *ircClient constructor*
- ~ircClient ()
- void doConnect ()

  *connects to the IRC server. Sets the whole thing in motion*
- void setOwnUser (ircUser *ownUser)

  *Sets our own user.*
- void sendChat (const char *msg, const char *channel)

  *Sends a chat message to a certain channel.*
- void sendChat (const char *msg)

  *Sends a chat message to the current channel.*
- void sendChat (const QString &msg)

  *Sends a chat message to the current channel.*
- void setUser (ircUser &user)

  *Sets our own user.*
- nickAndStatus getOwnNick () const

  *getOwnNick*
- void setOwnNick (const QString &newNick)

  *Sets our own nickname.*

### Private Member Functions

- void init ()

  *Initializes the IRC Client. Does not yet connect!*
- void send (const QString data)

  *Sends a message to the IRC server.*
- void handleCommand (const QString &sender, const QStringList &command)

  *Handles a command sent from the irc server.*
- void sendPong (const QString packet)

  *Replies to the server ping.*

### Private Attributes

- QString hostname

  *The hostname to which the IRC client has connected.*
- int port

  *The port on which the IRC client has connected.*
- QString channel

  *The channel the IRC client joined.*
- bool motdReceived

  *If the server has sent us the Message Of The Day.*
- QTcpSocket * sock

  *The socket communicating with the IRC server.*
- QBuffer * buffer

  *Buffer for receiving messages.*
- clsLog log

  *Log class for easy debugging.*
- nickAndStatus ownNickAndStatus
- ircUser ownUser
- bool connected
- bool connecting
- QTimer * connectTimer
- QTimer * timeOutChecker
- QTime lastPing
- int timeBetweenPings
- bool suppressDisconnectSignal

### 7.23.1 Detailed Description

The ircClient class.

**Author**

Koen Beckers (K-4U) My beautiful IRC Client.

Definition at line 27 of file ircclient.h.

### 7.23.2 Constructor & Destructor Documentation

#### 7.23.2.1 server::ircClient::ircClient ( const QString & *hostname,* const QString & *channel,* const QString & *nick,* const int *port* = PORT )

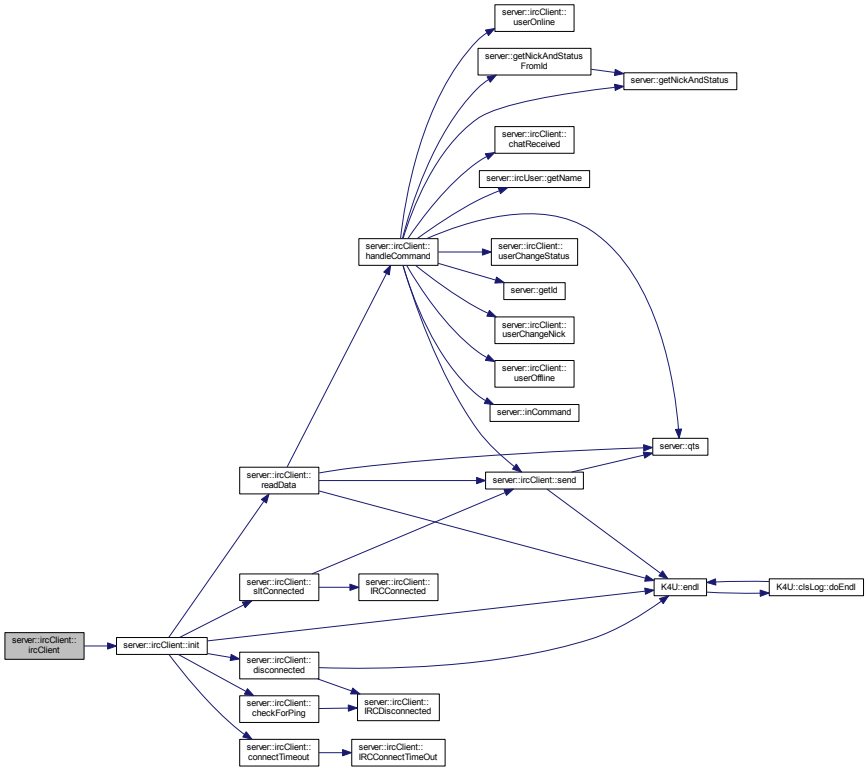
ircClient constructor

**Parameters**

| | |
|---|---|
| *hostname* | The IRC host to connect to |
| *port* | The IRC port to connect to (if not standard) |
| *channel* | The channel to join |
| *nick* | The nick to use |

Definition at line 8 of file ircclient.cpp.

Here is the call graph for this function:




#### 7.23.2.2 server::ircClient::~ircClient ( )

Definition at line 55 of file ircclient.cpp.

### 7.23.3 Member Function Documentation

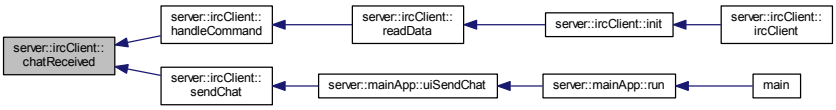
#### 7.23.3.1 void server::ircClient::chatReceived ( const QString *channel,* const nickAndStatus *user,* const QString *message* ) [signal]

Emitted when a chatmessage has been received.

**Parameters**

| | |
|---|---|
| *channel* | The channel in which the message has been received |
| *user* | The nick and status of the user sending the message |
| *message* | The message itself |

Here is the caller graph for this function:

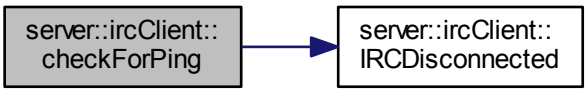



#### 7.23.3.2 void server::ircClient::checkForPing ( ) [slot]

Runs every CHECK_PING_INTERVAL to check if the last ping was there.

Definition at line 256 of file ircclient.cpp.

Here is the call graph for this function:

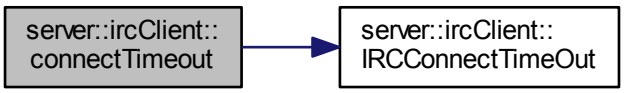



Here is the caller graph for this function:

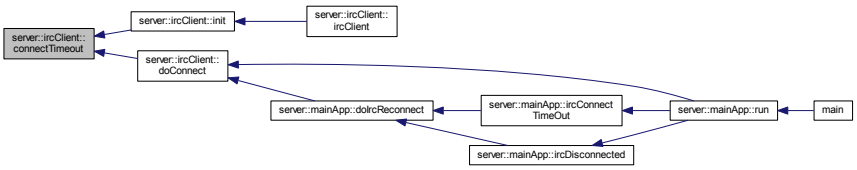
#### 7.23.3.3 void server::ircClient::connectTimeout ( ) [slot]

Definition at line 248 of file ircclient.cpp.

Here is the call graph for this function:



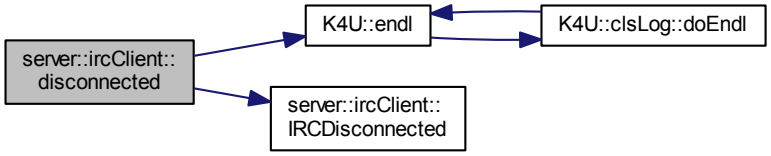

Here is the caller graph for this function:



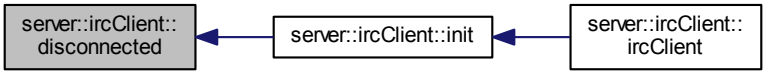

#### 7.23.3.4 void server::ircClient::disconnected ( ) [slot]

Definition at line 46 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

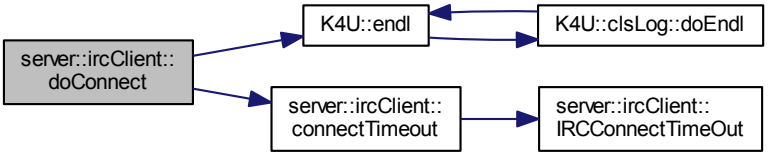


#### 7.23.3.5 void server::ircClient::doConnect ( )

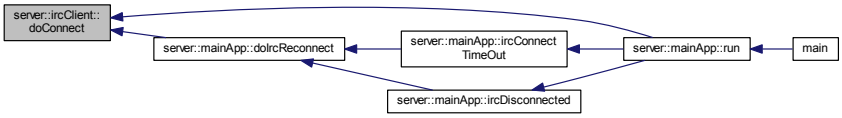
connects to the IRC server. Sets the whole thing in motion

Definition at line 60 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.23.3.6 nickAndStatus server::ircClient::getOwnNick ( ) const

getOwnNick

**Returns**

Our own nickname. At that moment known to us.

Definition at line 197 of file ircclient.cpp.

#### 7.23.3.7 void server::ircClient::handleCommand ( const QString & *sender*, const QStringList & *command* ) [private]

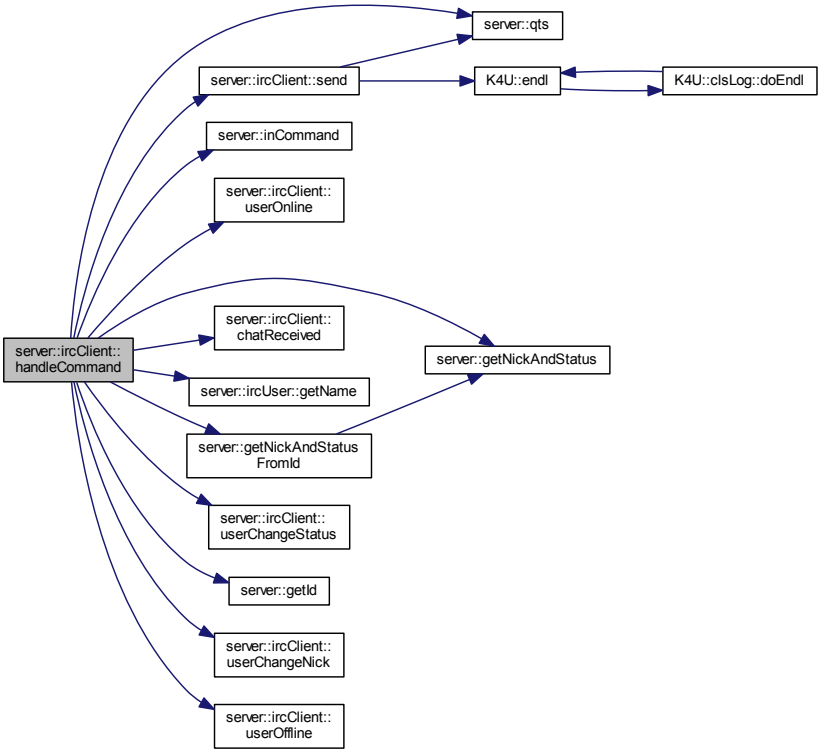
Handles a command sent from the irc server.

**Parameters**

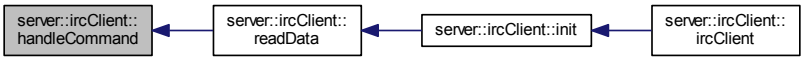
| | |
|---|---|
| *sender* | The IRC-ID that send the command |
| *command* | A string list containing the command |

Definition at line 94 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

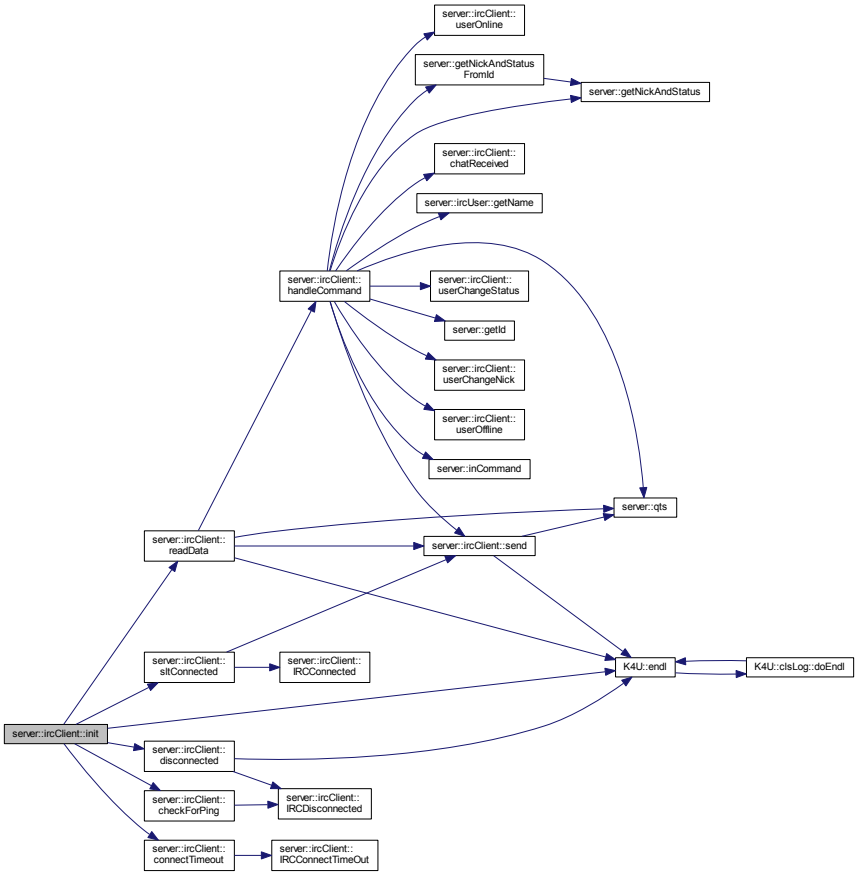



#### 7.23.3.8 void server::ircClient::init ( ) [private]

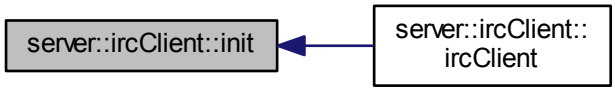
Initializes the IRC Client. Does not yet connect!

Definition at line 22 of file ircclient.cpp.

Here is the call graph for this function:



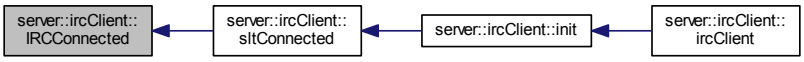

Here is the caller graph for this function:




#### 7.23.3.9 void server::ircClient::IRCConnected ( ) [signal]

Emitted when the connection has been established.

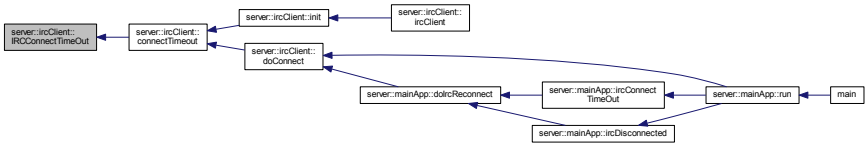
Here is the caller graph for this function:




#### 7.23.3.10 void server::ircClient::IRCConnectTimeOut ( ) [signal]

Emitted when a connection cannot be made after MAX_CONNECT_TIMEOUT_IN_MS.

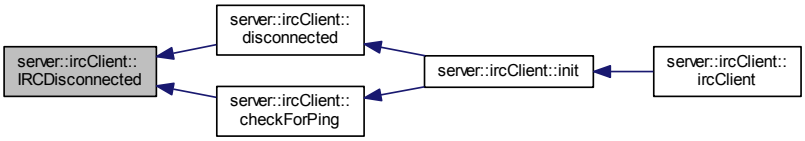
Here is the caller graph for this function:




#### 7.23.3.11 void server::ircClient::IRCDisconnected ( QString *reason* ) [signal]

Emitted when the connection was cut!

Here is the caller graph for this function:



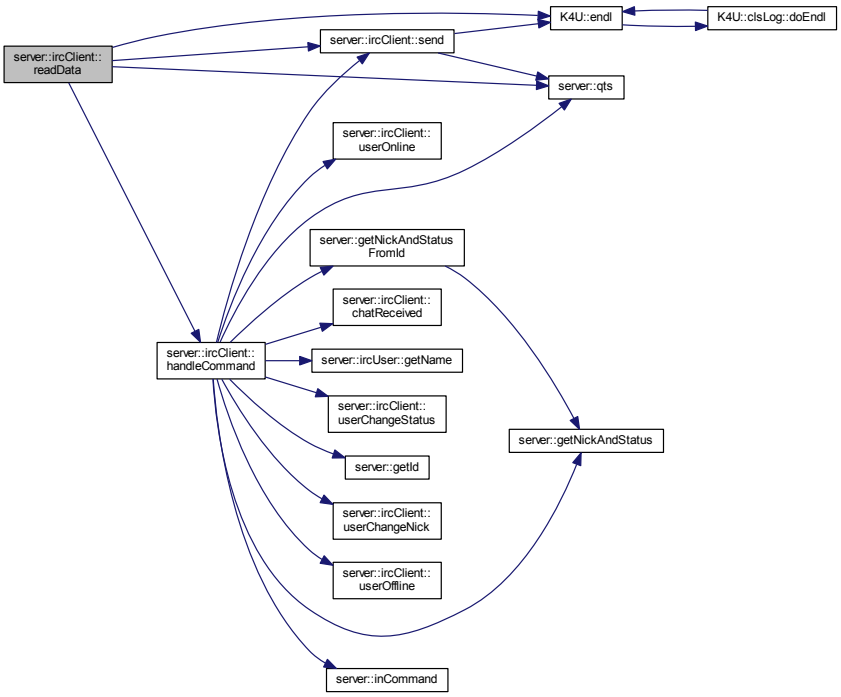

#### 7.23.3.12 void server::ircClient::readData ( ) [slot]

Reads data from the IRC Socket.

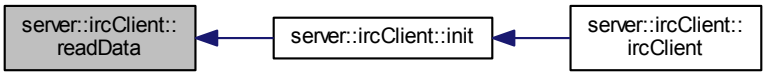
**Todo** Error handling

Definition at line 218 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.23.3.13 void server::ircClient::send ( const QString *data* ) [private]

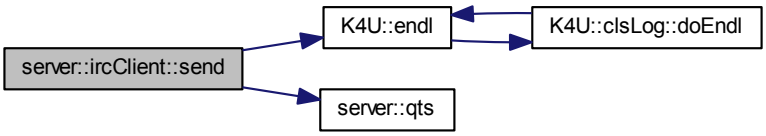
Sends a message to the IRC server.

**Parameters**

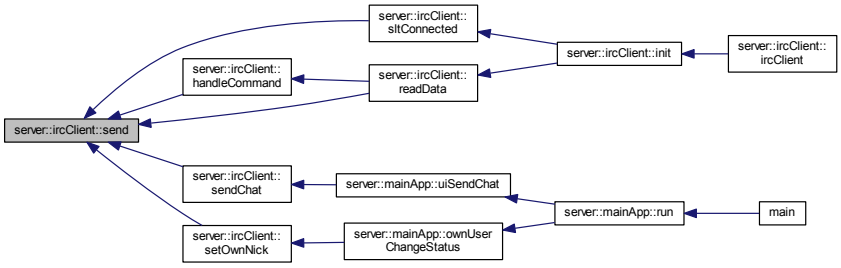
| | |
|---|---|
| *data* | The data to send |

Definition at line 207 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.23.3.14 void server::ircClient::sendChat ( const char * *msg*, const char * *channel* )

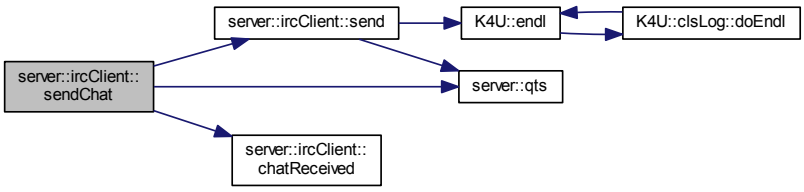
Sends a chat message to a certain channel.

**Parameters**

| | |
|---|---|
| *msg* | The message to send |
| *channel* | The channel to send it to |

Definition at line 174 of file ircclient.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

#### 7.23.3.15 void server::ircClient::sendChat ( const char * *msg* )

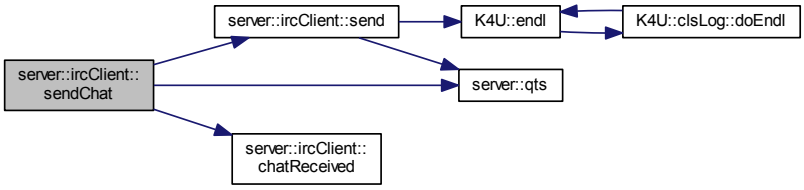
Sends a chat message to the current channel.

**Parameters**

| | |
|---|---|
| *msg* | The message to send |

Definition at line 181 of file ircclient.cpp.

Here is the call graph for this function:




#### 7.23.3.16 void server::ircClient::sendChat ( const QString & *msg* )

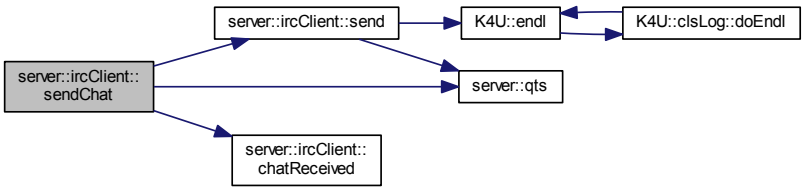
Sends a chat message to the current channel.

**Parameters**

| | |
|---|---|
| *msg* | The message to send |

Definition at line 187 of file ircclient.cpp.

Here is the call graph for this function:



#### 7.23.3.17 void server::ircClient::sendPong ( const QString *packet* ) [private]

Replies to the server ping.

**Parameters**

| | |
|---|---|
| *packet* | |

#### 7.23.3.18 void server::ircClient::setOwnNick ( const QString & *newNick* )

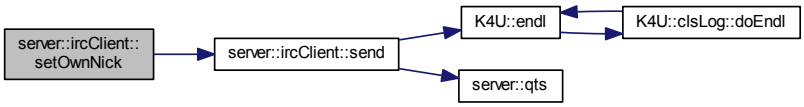
Sets our own nickname.

**Parameters**

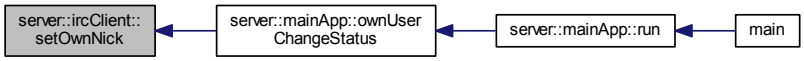
| | |
|---|---|
| *newNick* | The nickname to set |

Definition at line 201 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.23.3.19 void server::ircClient::setOwnUser ( ircUser * *ownUser* )

Sets our own user.

**Parameters**

| | |
|---|---|
| *ownUser* | The user to set as ourselfes |

Definition at line 75 of file ircclient.cpp.

Here is the caller graph for this function:

#### 7.23.3.20 void server::ircClient::setUser ( ircUser & *user* )

Sets our own user.

**Parameters**

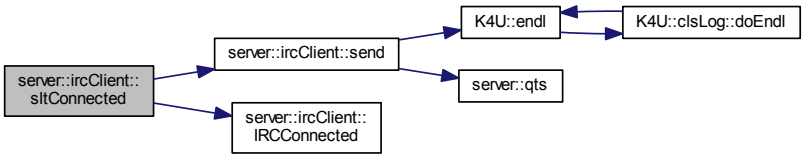
| | |
|---|---|
| *user* | Our new user. |

Definition at line 193 of file ircclient.cpp.

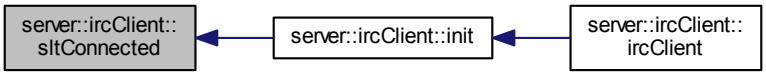
#### 7.23.3.21 void server::ircClient::sltConnected ( ) [slot]

Definition at line 79 of file ircclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.23.3.22 void server::ircClient::startupComplete ( ) [signal]

Emitted when the client is logged in. MOTD has been received.

#### 7.23.3.23 void server::ircClient::userChangeNick ( const nickAndStatus *oldNickName*, const nickAndStatus *newNickName*, const QString *id* ) [signal]

Emitted when a user changes his nickname (nick|Status)

**Parameters**

| | |
|---|---|
| *oldNickName* | The old nick and status this user had |
| *newNickName* | The new nick and status this user now has |
| *id* | His IRC-ID. The only thing probably that remained the same. |

**Warning**

Not to be confused with userChangeStatus!

Here is the caller graph for this function:

#### 7.23.3.24 void server::ircClient::userChangeStatus ( const nickAndStatus *oldNick*, const nickAndStatus *newNick*, const QString *id* ) [signal]

Emitted when a user changes his status (nick|Status)

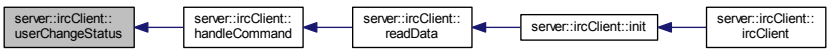
**Parameters**

| | |
|---|---|
| *oldNick* | The old nick and status this user had. |
| *newNick* | The new nick and status this user now has |
| *id* | His IRC-ID. The only thing probably that remained the same. |

**Warning**

Not to be confused with userChangeNick!

Here is the caller graph for this function:



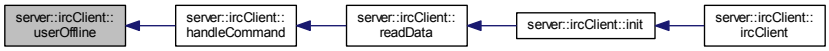

#### 7.23.3.25 void server::ircClient::userOffline ( const nickAndStatus *nickName*, const QString *id*, const QString *channel* ) [signal]

Emitted when a user leaves the room with a PART message.

**Parameters**

| | |
|---|---|
| *nickName* | The nick and status the user had when he left |
| *id* | The IRC-ID of this user |
| *channel* | The channel which he left |

Here is the caller graph for this function:



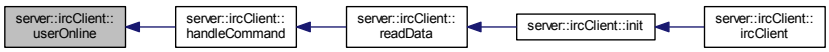

#### 7.23.3.26 void server::ircClient::userOnline ( const nickAndStatus *nickName*, const QString *id*, const QString *channel* ) [signal]

Emitted when a user comes online with a JOIN message.

**Parameters**

| | |
|---|---|
| *nickName* | The nick and status the user has. |
| *id* | The IRC-ID of this user |
| *channel* | The channel in which he has joined |

Here is the caller graph for this function:




### 7.23.4 Member Data Documentation

#### 7.23.4.1 QBuffer* server::ircClient::buffer [private]

Buffer for receiving messages.

Definition at line 219 of file ircclient.h.

#### 7.23.4.2 QString server::ircClient::channel [private]

The channel the IRC client joined.

Definition at line 205 of file ircclient.h.

#### 7.23.4.3 bool server::ircClient::connected [private]

Definition at line 228 of file ircclient.h.

#### 7.23.4.4 bool server::ircClient::connecting [private]

Definition at line 229 of file ircclient.h.

#### 7.23.4.5 QTimer* server::ircClient::connectTimer [private]

Definition at line 231 of file ircclient.h.

#### 7.23.4.6 QString server::ircClient::hostname [private]

The hostname to which the IRC client has connected.

Definition at line 197 of file ircclient.h.

#### 7.23.4.7 QTime server::ircClient::lastPing [private]

Definition at line 233 of file ircclient.h.

#### 7.23.4.8 clsLog server::ircClient::log [private]

Log class for easy debugging.

Definition at line 224 of file ircclient.h.

#### 7.23.4.9 bool server::ircClient::motdReceived [private]

If the server has sent us the Message Of The Day.

Definition at line 209 of file ircclient.h.

#### 7.23.4.10 nickAndStatus server::ircClient::ownNickAndStatus [private]

Definition at line 226 of file ircclient.h.

#### 7.23.4.11 ircUser server::ircClient::ownUser [private]

Definition at line 227 of file ircclient.h.

#### 7.23.4.12 int server::ircClient::port [private]

The port on which the IRC client has connected.

Definition at line 201 of file ircclient.h.

#### 7.23.4.13 QTcpSocket* server::ircClient::sock [private]

The socket communicating with the IRC server.

Definition at line 214 of file ircclient.h.

#### 7.23.4.14 bool server::ircClient::suppressDisconnectSignal [private]

Definition at line 235 of file ircclient.h.

#### 7.23.4.15 int server::ircClient::timeBetweenPings [private]

Definition at line 234 of file ircclient.h.

#### 7.23.4.16 QTimer* server::ircClient::timeOutChecker [private]

Definition at line 232 of file ircclient.h.

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

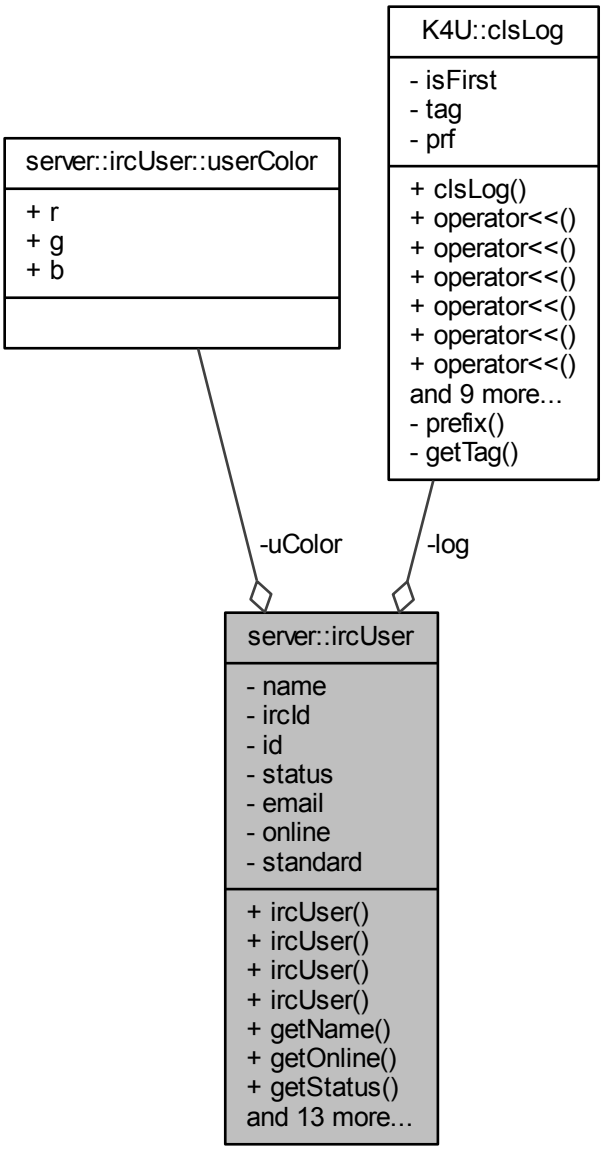
- C++/customIrcServer/ircclient.h
- C++/customIrcServer/ircclient.cpp

## 7.24 server::ircUser Class Reference

The ircUser class.

```
#include <ircuser.h>
```

Collaboration diagram for server::ircUser:




## Classes

- struct userColor

  *The userColor struct As a replacement for the QColor type.*

## Public Member Functions

- ircUser (const nickAndStatus name, const QString ircId)

  *ircUser constructor*
- ircUser (const QString name, const bool standard)

*ircUser constructor*

- ircUser (const nickAndStatus name)

  *ircUser constructor*

- ircUser ()
- QString getName () const
- bool getOnline () const
- QString getStatus () const
- bool isStandard () const
- QVariantMap toVariantMap () const

  *Used to convert a user for JSON transfer.*

- QString getIrcId () const

  *getIrcId*

- int getId () const

  *getId*

- void setOnline (const bool isOnline)
- void setStatus (const QString &newStatus)
- void setIrcId (const QString &newId)
- void setId (const int &newId)
- void setNick (const QString &newNick)
- void setColor (const int r, const int g, const int b)
- void setColor (const userColor &newColor)
- void setEmail (const QString &email)
- ircUser * operator= (const ircUser *user)

### Private Attributes

- QString name
- QString ircId
- int id
- QString status
- QString email
- bool online
- bool standard
- userColor uColor
- clsLog log

### Friends

- clsLog & operator<< (clsLog &log, const ircUser *user)

### 7.24.1 Detailed Description

The ircUser class.

**Author**

Koen Beckers (K-4U) Contains an IRC User There are more than one way to create a user. Please look at the options carefully!

Definition at line 19 of file ircuser.h.

### 7.24.2 Constructor & Destructor Documentation

#### 7.24.2.1 server::ircUser::ircUser ( const nickAndStatus *name*, const QString *ircId* )

ircUser constructor

**Parameters**

| | |
|---|---|
| *name* | The nick and status of this user. |
| *ircId* | The IRC-id of this user |

Definition at line 5 of file ircuser.cpp.

#### 7.24.2.2 server::ircUser::ircUser ( const QString *name*, const bool *standard* )

ircUser constructor

**Parameters**

| | |
|---|---|
| *name* | The nickname of this user |
| *standard* | If the user is a regular user Usually this is used when loading users from a settings file. |

Definition at line 49 of file ircuser.cpp.

#### 7.24.2.3 server::ircUser::ircUser ( const nickAndStatus *name* )

ircUser constructor

**Parameters**

| | |
|---|---|
| *name* | The nickname of this user Use of this constructor is highly discouraged as it cannot tell everything about a user! |

Definition at line 27 of file ircuser.cpp.

#### 7.24.2.4 server::ircUser::ircUser ( )

Definition at line 77 of file ircuser.cpp.

### 7.24.3 Member Function Documentation

#### 7.24.3.1 int server::ircUser::getId ( ) const

getId

**Returns**

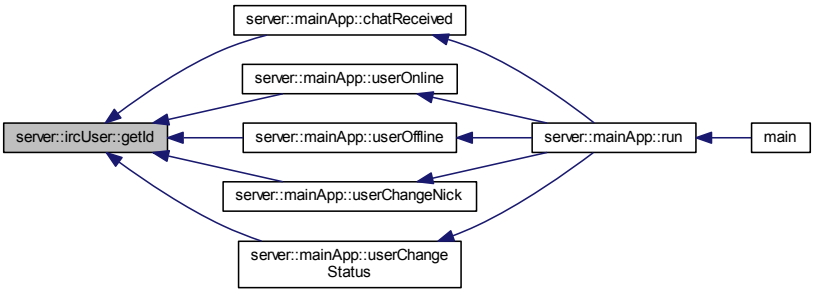
The ID of this user in the database

**Warning**

Do not confuse with getIrcId(), which returns the IRC-ID

Definition at line 162 of file ircuser.cpp.

Here is the caller graph for this function:




#### 7.24.3.2 QString server::ircUser::getIrcId ( ) const

getIrcId

**Returns**

The IRC-ID of this user.

**Warning**

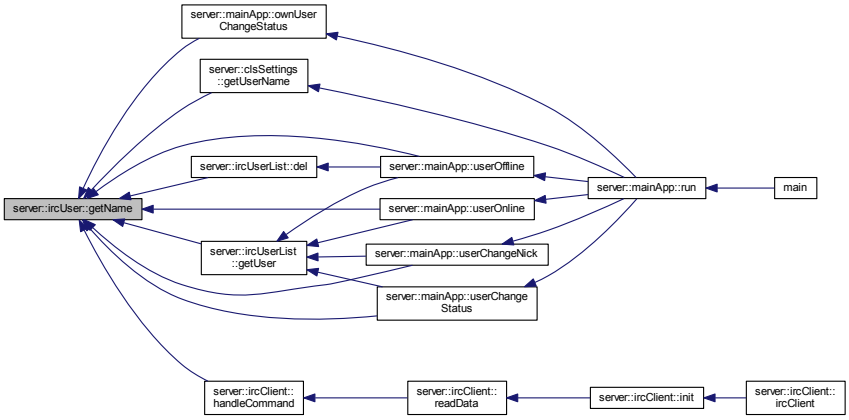
Do not confuse with getId(), which returns the user Id in the database

Definition at line 158 of file ircuser.cpp.

#### 7.24.3.3 QString server::ircUser::getName ( ) const

Definition at line 91 of file ircuser.cpp.

Here is the caller graph for this function:



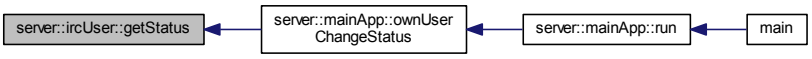
#### 7.24.3.4 bool server::ircUser::getOnline ( ) const

Definition at line 127 of file ircuser.cpp.

#### 7.24.3.5 QString server::ircUser::getStatus ( ) const

Definition at line 95 of file ircuser.cpp.

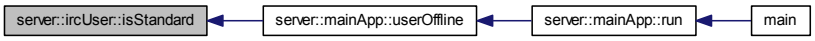
Here is the caller graph for this function:




#### 7.24.3.6 bool server::ircUser::isStandard ( ) const

Definition at line 99 of file ircuser.cpp.

Here is the caller graph for this function:




#### 7.24.3.7 ircUser * server::ircUser::operator= ( const ircUser * *user* )

Definition at line 284 of file ircuser.cpp.

#### 7.24.3.8 void server::ircUser::setColor ( const int *r*, const int *g*, const int *b* )

Definition at line 143 of file ircuser.cpp.

Here is the caller graph for this function:

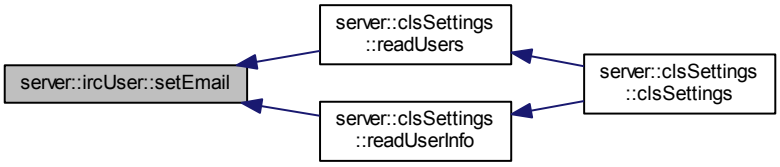
#### 7.24.3.9 void server::ircUser::setColor ( const userColor & *newColor* )

Definition at line 149 of file ircuser.cpp.

#### 7.24.3.10 void server::ircUser::setEmail ( const QString & *email* )

Definition at line 153 of file ircuser.cpp.

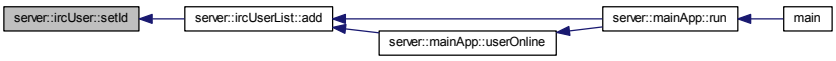
Here is the caller graph for this function:




#### 7.24.3.11 void server::ircUser::setId ( const int & *newId* )

Definition at line 166 of file ircuser.cpp.

Here is the caller graph for this function:



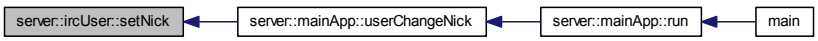

#### 7.24.3.12 void server::ircUser::setIrcId ( const QString & *newId* )

Definition at line 135 of file ircuser.cpp.

#### 7.24.3.13 void server::ircUser::setNick ( const QString & *newNick* )

Definition at line 139 of file ircuser.cpp.

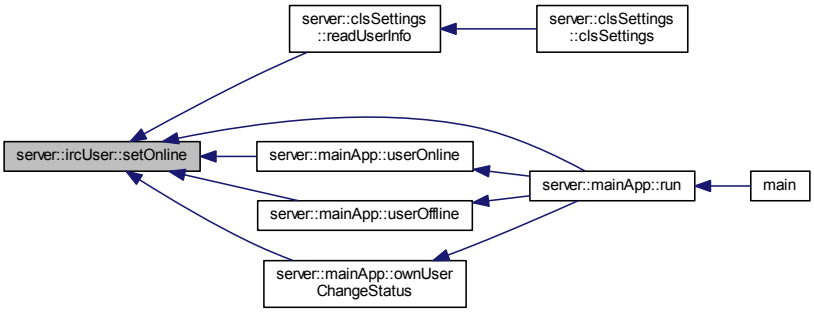
Here is the caller graph for this function:




#### 7.24.3.14 void server::ircUser::setOnline ( const bool *isOnline* )

Definition at line 123 of file ircuser.cpp.

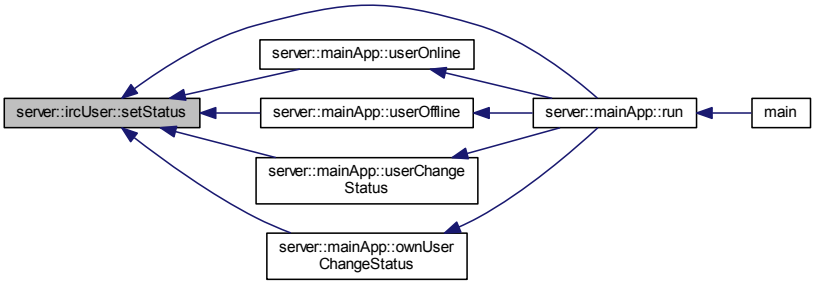
Here is the caller graph for this function:




#### 7.24.3.15 void server::ircUser::setStatus ( const QString & *newStatus* )

Definition at line 131 of file ircuser.cpp.

Here is the caller graph for this function:




#### 7.24.3.16 QVariantMap server::ircUser::toVariantMap ( ) const

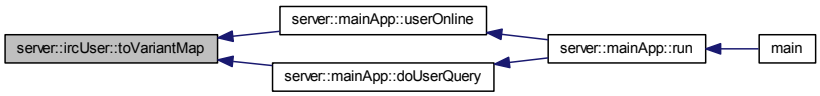
Used to convert a user for JSON transfer.

**Returns**

A variant map with all of the users´ data.

Definition at line 103 of file ircuser.cpp.

Here is the caller graph for this function:



### 7.24.4 Friends And Related Function Documentation

#### 7.24.4.1 clsLog& operator<< ( clsLog & *log*, const ircUser * *user* ) [friend]

Definition at line 170 of file ircuser.cpp.

### 7.24.5 Member Data Documentation

#### 7.24.5.1 QString server::ircUser::email [private]

Definition at line 100 of file ircuser.h.

#### 7.24.5.2 int server::ircUser::id [private]

Definition at line 98 of file ircuser.h.

#### 7.24.5.3 QString server::ircUser::ircId [private]

Definition at line 97 of file ircuser.h.

#### 7.24.5.4 clsLog server::ircUser::log [private]

Definition at line 106 of file ircuser.h.

#### 7.24.5.5 QString server::ircUser::name [private]

Definition at line 96 of file ircuser.h.

#### 7.24.5.6 bool server::ircUser::online [private]

Definition at line 101 of file ircuser.h.

#### 7.24.5.7 bool server::ircUser::standard [private]

Definition at line 102 of file ircuser.h.

#### 7.24.5.8 QString server::ircUser::status [private]

Definition at line 99 of file ircuser.h.

#### 7.24.5.9 userColor server::ircUser::uColor [private]

Definition at line 104 of file ircuser.h.

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

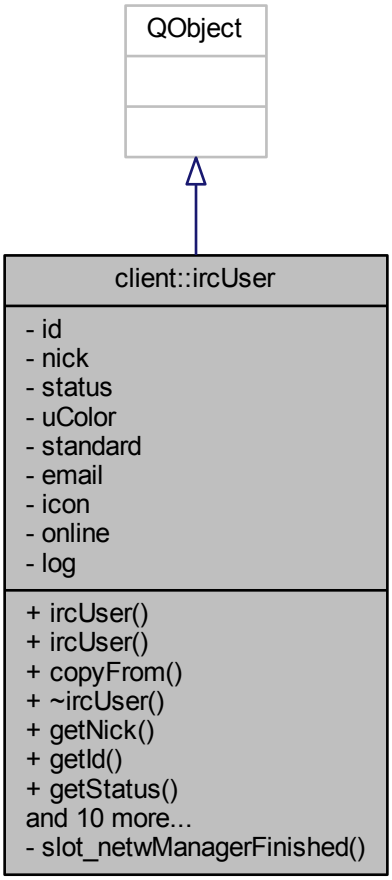
- C++/customIrcServer/ircuser.h
- C++/customIrcServer/ircuser.cpp

## 7.25 client::ircUser Class Reference

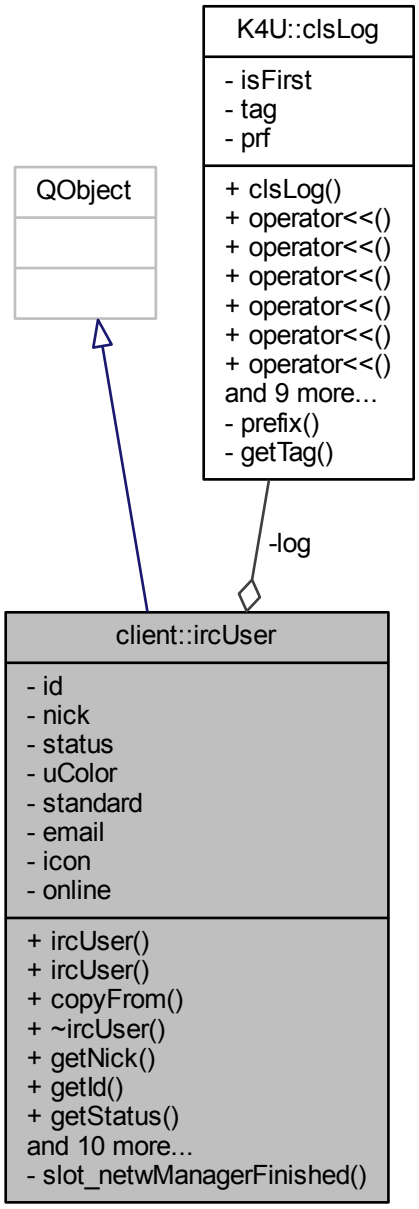
The ircUser class.

```
#include <ircuser.h>
```

Inheritance diagram for client::ircUser:



Collaboration diagram for client::ircUser:




## Signals

- void iconChanged () const

## Public Member Functions

- ircUser (QVariant jsonData)

  *ircUser constructor taking a QVariant for the jsonData*

- ircUser (qint32 id, QString nick, QString status, QColor uColor, bool standard)

  *ircUser constructor taking all the users data*
- void copyFrom (ircUser *userToCopy)
- ~ircUser ()
- const QString getNick () const
- qint32 getId () const
- const QString getStatus () const
- const QColor getColor () const
- const QString getEmail () const
- const QPixmap getIcon () const
- bool getStandard () const
- bool getOnline () const
- void setNick (const QString &newNick)
- void setStatus (const QString &newStatus)
- void setColor (const QColor &newColor)
- void setEmail (const QString &email)
- void setOnline (const bool isOnline)

### Private Slots

- void slot_netwManagerFinished (QNetworkReply *reply)

### Private Attributes

- qint32 id
- QString nick
- QString status
- QColor uColor
- bool standard
- QString email
- QPixmap icon
- bool online
- clsLog log

## 7.25.1 Detailed Description

The ircUser class.

**Author**

Koen Beckers (K-4U) Contains an IRC User There are more than one way to create a user. Please look at the options carefully!

Definition at line 23 of file ircuser.h.

## 7.25.2 Constructor & Destructor Documentation

### 7.25.2.1 client::ircUser::ircUser ( QVariant *jsonData* ) [explicit]

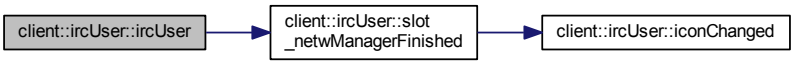
ircUser constructor taking a QVariant for the jsonData

**Parameters**

| | |
|---|---|
| *jsonData* | The QVariant containing the JSON data |

Definition at line 10 of file ircuser.cpp.

Here is the call graph for this function:




#### 7.25.2.2 client::ircUser::ircUser ( qint32 *id*, QString *nick*, QString *status*, QColor *uColor*, bool *standard* ) [explicit]

ircUser constructor taking all the users data

**Parameters**

| | |
|---|---|
| *id* | The DB-ID of this user |
| *nick* | The nickname of this user |
| *status* | The status of this user |
| *uColor* | The user color |
| *standard* | Determines if he should be removed or left in the list when he leaves |

Definition at line 37 of file ircuser.cpp.

#### 7.25.2.3 client::ircUser::∼ircUser ( )

Definition at line 6 of file ircuser.cpp.

### 7.25.3 Member Function Documentation

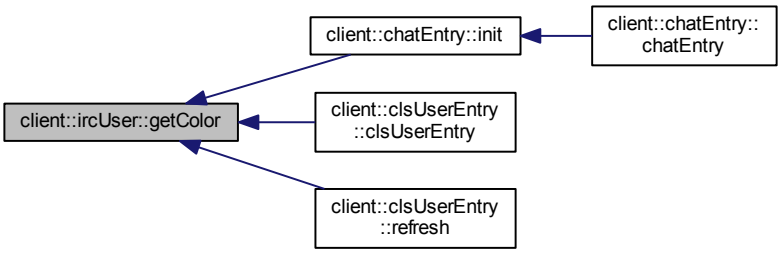
#### 7.25.3.1 void client::ircUser::copyFrom ( ircUser * *userToCopy* )

Definition at line 46 of file ircuser.cpp.

#### 7.25.3.2 const QColor client::ircUser::getColor ( ) const

Definition at line 66 of file ircuser.cpp.

Here is the caller graph for this function:

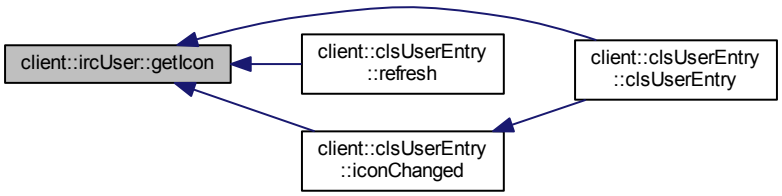


#### 7.25.3.3 const QString client::ircUser::getEmail ( ) const

#### 7.25.3.4 const QPixmap client::ircUser::getIcon ( ) const

Definition at line 70 of file ircuser.cpp.

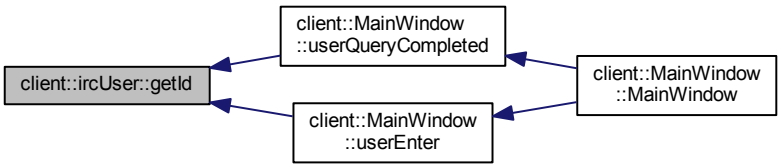
Here is the caller graph for this function:




#### 7.25.3.5 qint32 client::ircUser::getId ( ) const

Definition at line 58 of file ircuser.cpp.

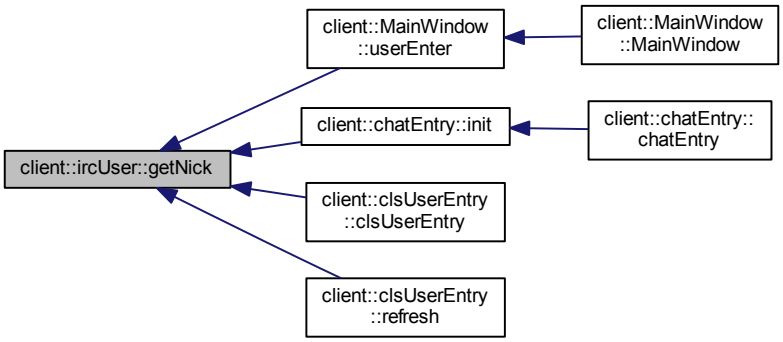
Here is the caller graph for this function:




#### 7.25.3.6 const QString client::ircUser::getNick ( ) const

Definition at line 53 of file ircuser.cpp.

Here is the caller graph for this function:



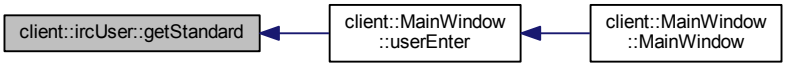

#### 7.25.3.7 bool client::ircUser::getOnline ( ) const

Definition at line 78 of file ircuser.cpp.

#### 7.25.3.8 bool client::ircUser::getStandard ( ) const

Definition at line 74 of file ircuser.cpp.

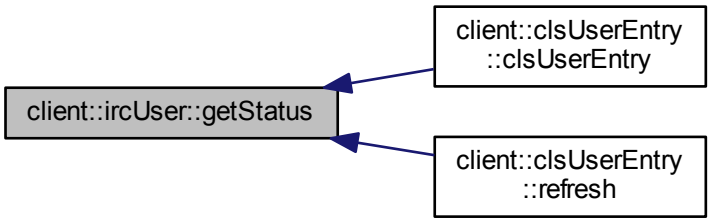
Here is the caller graph for this function:




#### 7.25.3.9 const QString client::ircUser::getStatus ( ) const

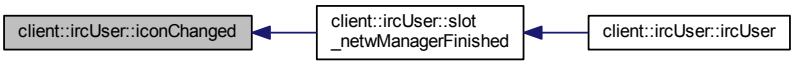
Definition at line 62 of file ircuser.cpp.

Here is the caller graph for this function:




#### 7.25.3.10 void client::ircUser::iconChanged ( ) const [signal]

Here is the caller graph for this function:




#### 7.25.3.11 void client::ircUser::setColor ( const QColor & *newColor* )

Definition at line 94 of file ircuser.cpp.

#### 7.25.3.12 void client::ircUser::setEmail ( const QString & *email* )

#### 7.25.3.13 void client::ircUser::setNick ( const QString & *newNick* )

Definition at line 82 of file ircuser.cpp.

#### 7.25.3.14 void client::ircUser::setOnline ( const bool *isOnline* )

Definition at line 112 of file ircuser.cpp.

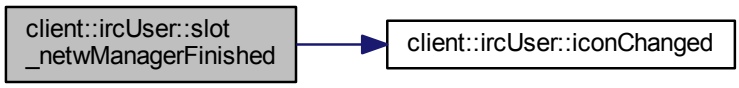
#### 7.25.3.15 void client::ircUser::setStatus ( const QString & *newStatus* )

Definition at line 86 of file ircuser.cpp.

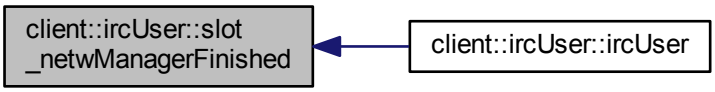
#### 7.25.3.16 void client::ircUser::slot_netwManagerFinished ( QNetworkReply * *reply* ) [private],[slot]

Definition at line 99 of file ircuser.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




### 7.25.4 Member Data Documentation

#### 7.25.4.1 QString client::ircUser::email [private]

Definition at line 77 of file ircuser.h.

#### 7.25.4.2 QPixmap client::ircUser::icon [private]

Definition at line 78 of file ircuser.h.

#### 7.25.4.3 qint32 client::ircUser::id [private]

Definition at line 72 of file ircuser.h.

#### 7.25.4.4 clsLog client::ircUser::log [private]

Definition at line 81 of file ircuser.h.

#### 7.25.4.5 QString client::ircUser::nick [private]

Definition at line 73 of file ircuser.h.

#### 7.25.4.6 bool client::ircUser::online [private]

Definition at line 79 of file ircuser.h.

#### 7.25.4.7 bool client::ircUser::standard [private]

Definition at line 76 of file ircuser.h.

#### 7.25.4.8 QString client::ircUser::status [private]

Definition at line 74 of file ircuser.h.

#### 7.25.4.9 QColor client::ircUser::uColor [private]

Definition at line 75 of file ircuser.h.

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

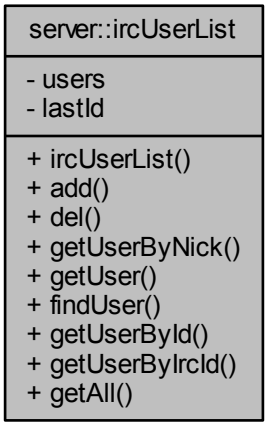
- C++/customIrcClient/ircuser.h
- C++/customIrcClient/ircuser.cpp

## 7.26 server::ircUserList Class Reference

Contains all the users.

`#include <ircuser.h>`

Collaboration diagram for server::ircUserList:




### Public Member Functions

- ircUserList ()
- void add (ircUser *userToAdd)

  *Adds a user to the list.*
- void del (ircUser *userToDel)

  *Removes a user from the list.*
- ircUser * getUserByNick (nickAndStatus nick)

*Finds a user based on his nick and status.*

- ircUser * getUser (const nickAndStatus nick, const QString ircId)

  *Finds a user based on IRC-ID or nick.*
- ircUser * findUser (ircUser &userToFind)

  *Finds a user based on an exact copy.*
- ircUser * getUserById (int id)

  *Gets a user by his database ID.*
- ircUser * getUserByIrcId (QString ircId)

  *Gets a user by his IRC-ID.*
- QVector< ircUser * > getAll () const

  *Get all the users. Used for foreach.*

### Private Attributes

- QVector< ircUser * > users
- int lastId

## 7.26.1 Detailed Description

Contains all the users.

**Author**

Koen Beckers (K-4U)

Definition at line 113 of file ircuser.h.

## 7.26.2 Constructor & Destructor Documentation

### 7.26.2.1 server::ircUserList::ircUserList ( )

Definition at line 177 of file ircuser.cpp.

## 7.26.3 Member Function Documentation

### 7.26.3.1 void server::ircUserList::add ( ircUser * *userToAdd* )

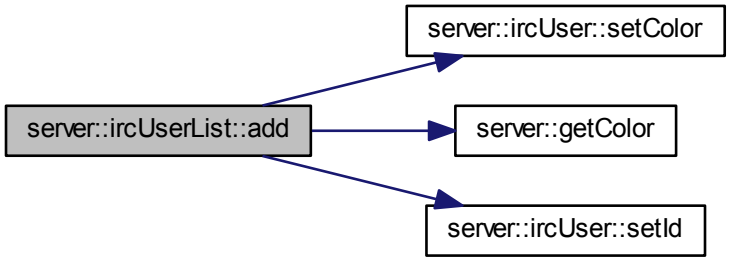
Adds a user to the list.

**Parameters**

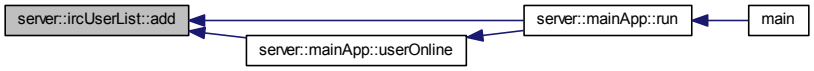
| | |
|---|---|
| *userToAdd* | |

Definition at line 233 of file ircuser.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.26.3.2 void server::ircUserList::del ( ircUser * *userToDel* )

Removes a user from the list.

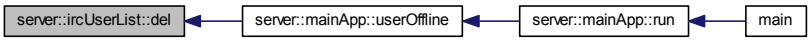
**Parameters**

| | |
|---|---|
| *userToDel* | |

Definition at line 240 of file ircuser.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:




#### 7.26.3.3 ircUser* server::ircUserList::findUser ( ircUser & *userToFind* )

Finds a user based on an exact copy.

**Parameters**

| | |
|---|---|
| *userToFind* | The user to find |

**Returns**

A pointer to the correct ircUser object if found. Else a pointer to 0x00

**Note**

Not sure why this function exists. If you have the user, why would you want to find it again?

#### 7.26.3.4 QVector< ircUser * > server::ircUserList::getAll ( ) const

Get all the users. Used for foreach.

**Returns**

A vector with pointers to all the users.

Definition at line 280 of file ircuser.cpp.

#### 7.26.3.5 ircUser * server::ircUserList::getUser ( const nickAndStatus *nick*, const QString *ircId* )

Finds a user based on IRC-ID or nick.

**Parameters**

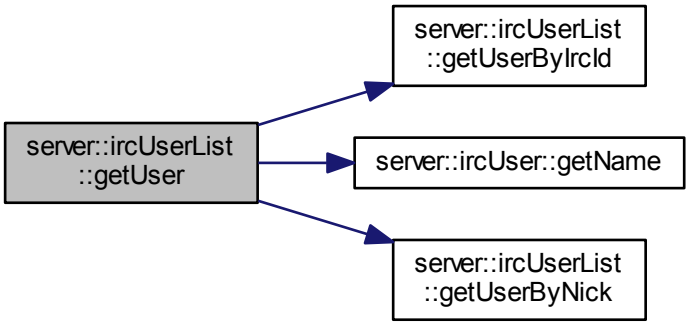
| | |
|---|---|
| *nick* | The nick and status to search for |
| *ircId* | The IRC-ID to search for |

**Returns**

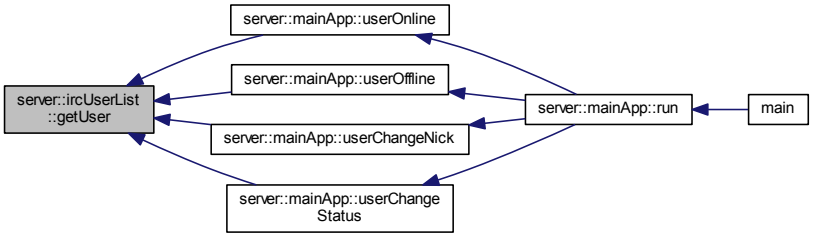
A pointer to the correct ircUser object if found. Else a pointer to 0x00

Definition at line 248 of file ircuser.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.26.3.6 ircUser* server::ircUserList::getUserById ( int *id* )

Gets a user by his database ID.

**Parameters**

| | |
|---|---|
| *id* | The ID to search for |

**Returns**

A pointer to the correct ircUser object if found. Else a pointer to 0x00

**Warning**

Do not confuse this with getUserByIrcId which searches for a user based on his IRC-ID!

#### 7.26.3.7 ircUser * server::ircUserList::getUserByIrcId ( QString *ircId* )

Gets a user by his IRC-ID.

**Parameters**

| | |
|---|---|
| *ircId* | The IRC-ID to search for |

**Returns**

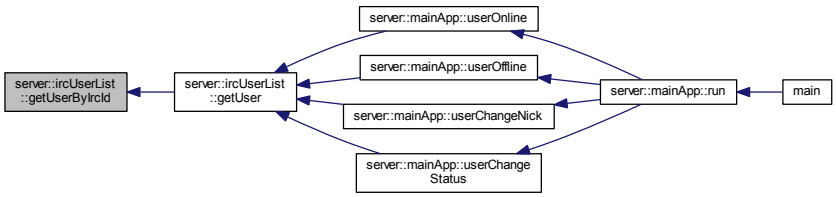
A pointer to the correct ircUser object if found. Else a pointer to 0x00

**Warning**

Do not confuse this with getUserById which searches for a user based on his database ID!

Definition at line 271 of file ircuser.cpp.

Here is the caller graph for this function:




#### 7.26.3.8 ircUser * server::ircUserList::getUserByNick ( nickAndStatus *nick* )

Finds a user based on his nick and status.

**Parameters**

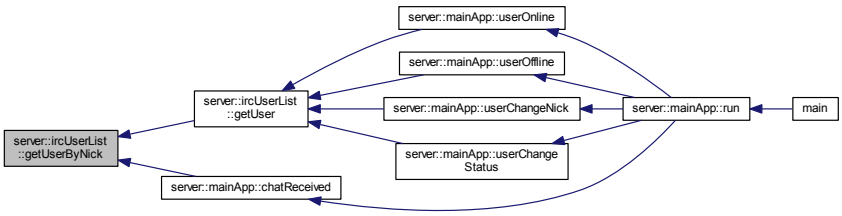
| | |
|---|---|
| *nick* | The nick and status to search for. |

**Returns**

A pointer to the correct ircUser object if found. Else a pointer to 0x00

Definition at line 262 of file ircuser.cpp.

Here is the caller graph for this function:



### 7.26.4 Member Data Documentation

#### 7.26.4.1 int server::ircUserList::lastId [private]

Definition at line 174 of file ircuser.h.

#### 7.26.4.2 QVector<ircUser*> server::ircUserList::users [private]

Definition at line 173 of file ircuser.h.

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

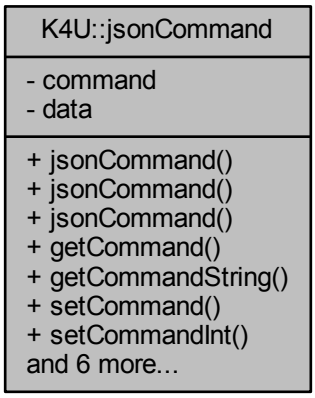
- C++/customIrcServer/ircuser.h
- C++/customIrcServer/ircuser.cpp

## 7.27 K4U::jsonCommand Class Reference

The jsonCommand class.

`#include <jsoncommand.h>`

Collaboration diagram for K4U::jsonCommand:




### Public Member Functions

- jsonCommand (QString json)

  *jsonCommand constructor taking a JSON string*
- jsonCommand (QVariantMap parent)

  *jsonCommand constructor taking a QVariantMap entry*
- jsonCommand (jsonCommands command)

  *jsonCommand constructor initializing an empty command*
- jsonCommands getCommand () const

  *getCommand*
- QString getCommandString () const

*getCommandString*

- void setCommand (const jsonCommands &newCommand)
- void setCommandInt (QString command)
- QVariant getData (const QString key)

  *Gets an item from the internal data map.*
- QVariantMap getData () const

  *Gets all data items.*
- void setData (const QVariantMap &newData)

  *Sets a new internal data map.*
- void addToData (const QString key, const QVariant &data)

  *Adds data to the internal data map.*
- jsonCommand operator= (jsonCommand *other)
- QString toJsonString () const

  *Transforms this object to a nice JSON string.*

### Private Attributes

- jsonCommands command
- QVariantMap data

### Friends

- clsLog & operator<< (clsLog &log, jsonCommand &command)
- clsLog & operator<< (clsLog &log, const jsonCommand &command)

## 7.27.1 Detailed Description

The jsonCommand class.

**Author**

Koen Beckers (K-4U)

Example command:

```
{
   "command": "OWNCHAT",
   "data": {
       "message":"Derp"
       }
}
```

Definition at line 40 of file jsoncommand.h.

## 7.27.2 Constructor & Destructor Documentation

### 7.27.2.1 K4U::jsonCommand::jsonCommand ( QString *json* )

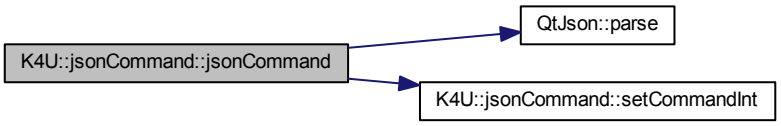
jsonCommand constructor taking a JSON string

**Parameters**

| | |
|---|---|
| *json* | A QString containing JSON |

Definition at line 5 of file jsoncommand.cpp.

Here is the call graph for this function:




#### 7.27.2.2 K4U::jsonCommand::jsonCommand ( QVariantMap *parent* )

jsonCommand constructor taking a QVariantMap entry

**Parameters**

| | |
|---|---|
| *parent* | The QVariantMap containing JSON data This will mostly be used when a JSON entry has been parsed by the JSON parser |

Definition at line 13 of file jsoncommand.cpp.

#### 7.27.2.3 K4U::jsonCommand::jsonCommand ( jsonCommands *command* )

jsonCommand constructor initializing an empty command

**Parameters**

| | |
|---|---|
| *command* | The command type to set |

Definition at line 19 of file jsoncommand.cpp.

### 7.27.3 Member Function Documentation

#### 7.27.3.1 void K4U::jsonCommand::addToData ( const QString *key*, const QVariant & *data* )

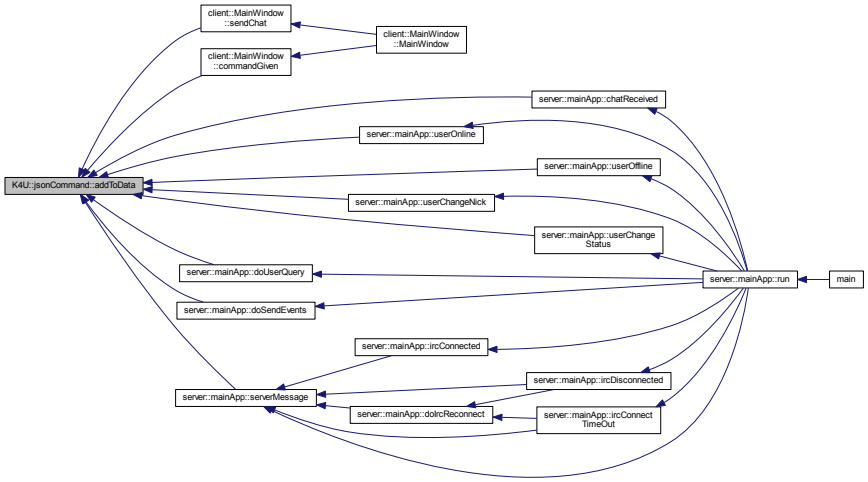
Adds data to the internal data map.

**Parameters**

| | |
|---|---|
| *key* | The key to add the data under |
| *data* | The data to add |

Definition at line 98 of file jsoncommand.cpp.

Here is the caller graph for this function:




#### 7.27.3.2 jsonCommands K4U::jsonCommand::getCommand ( ) const

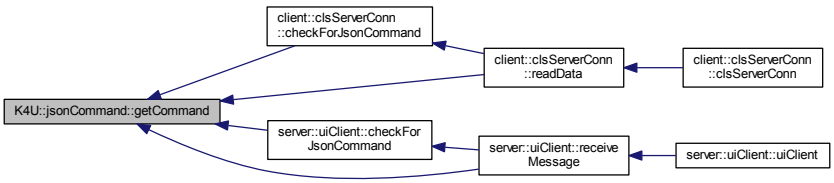
getCommand

**Returns**

A jsonCommands object containing the current command type

Definition at line 24 of file jsoncommand.cpp.

Here is the caller graph for this function:




#### 7.27.3.3 QString K4U::jsonCommand::getCommandString ( ) const

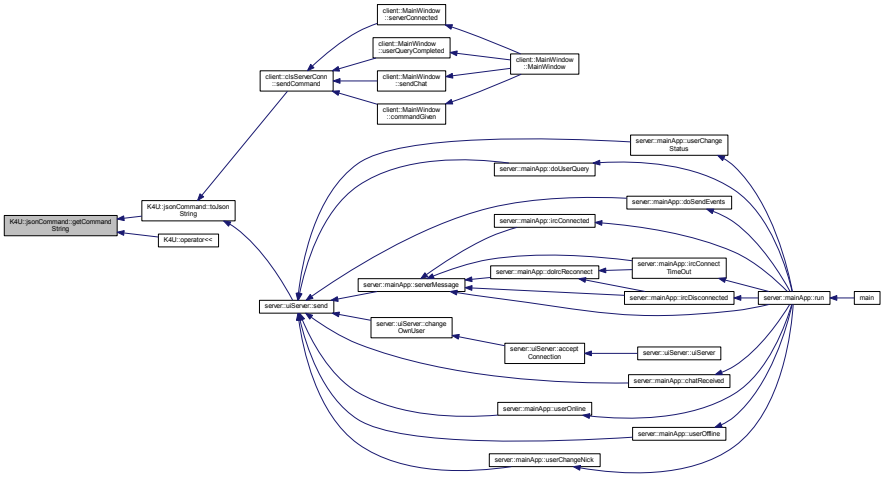
getCommandString

**Returns**

A QString containing the current command type

Definition at line 28 of file jsoncommand.cpp.

Here is the caller graph for this function:




#### 7.27.3.4 QVariant K4U::jsonCommand::getData ( const QString *key* )

Gets an item from the internal data map.

**Parameters**

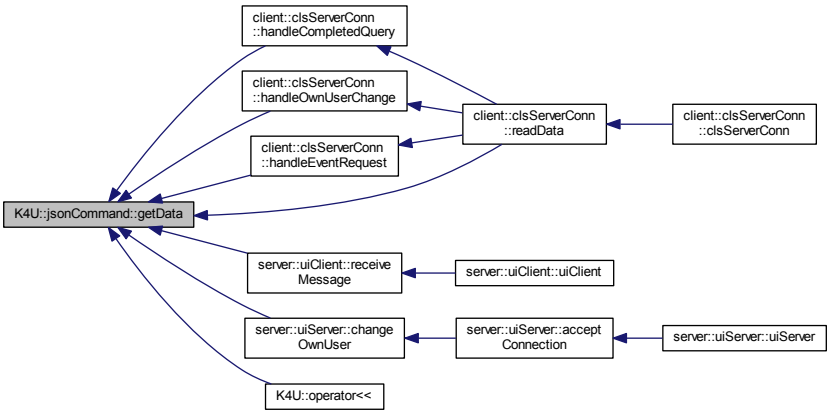
| | |
|---|---|
| *key* | The key to get |

**Returns**

A QVariant Object containing the data

Definition at line 89 of file jsoncommand.cpp.

Here is the caller graph for this function:



#### 7.27.3.5 QVariantMap K4U::jsonCommand::getData ( ) const

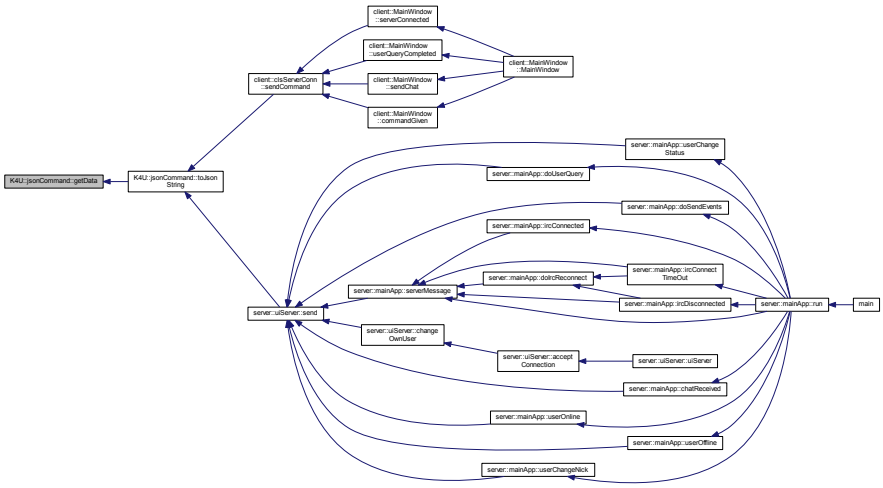
Gets all data items.

**Returns**

A QVariantMap containg all the data

Definition at line 85 of file jsoncommand.cpp.

Here is the caller graph for this function:




#### 7.27.3.6 jsonCommand K4U::jsonCommand::operator= ( jsonCommand * *other* )

Definition at line 123 of file jsoncommand.cpp.

#### 7.27.3.7 void K4U::jsonCommand::setCommand ( const jsonCommands & *newCommand* )

Definition at line 80 of file jsoncommand.cpp.

#### 7.27.3.8 void K4U::jsonCommand::setCommandInt ( QString *command* )

Definition at line 59 of file jsoncommand.cpp.

Here is the caller graph for this function:

#### 7.27.3.9 void K4U::jsonCommand::setData ( const QVariantMap & *newData* )

Sets a new internal data map.

**Parameters**

| | |
|---|---|
| *newData* | The new map to set |

Definition at line 94 of file jsoncommand.cpp.

#### 7.27.3.10 QString K4U::jsonCommand::toJsonString ( ) const

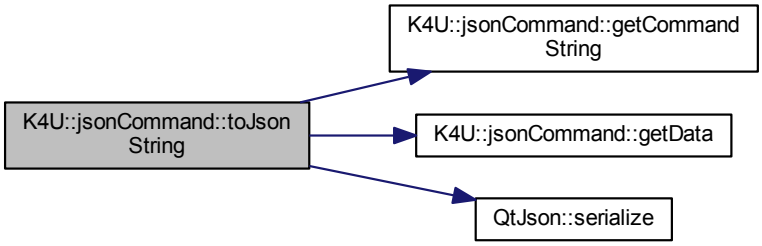
Transforms this object to a nice JSON string.

**Returns**

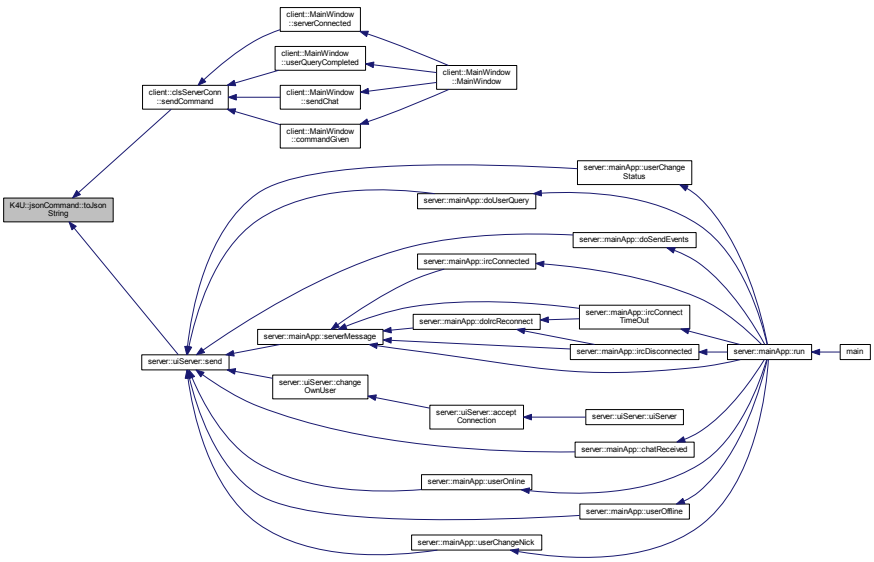
The JSON string

Definition at line 102 of file jsoncommand.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:



### 7.27.4 Friends And Related Function Documentation

#### 7.27.4.1 clsLog& operator<< ( clsLog & *log*, jsonCommand & *command* ) [friend]

Definition at line 110 of file jsoncommand.cpp.

#### 7.27.4.2 clsLog& operator<< ( clsLog & *log*, const jsonCommand & *command* ) [friend]

Definition at line 117 of file jsoncommand.cpp.

### 7.27.5 Member Data Documentation

#### 7.27.5.1 jsonCommands K4U::jsonCommand::command [private]

Definition at line 111 of file jsoncommand.h.

#### 7.27.5.2 QVariantMap K4U::jsonCommand::data [private]

Definition at line 112 of file jsoncommand.h.

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

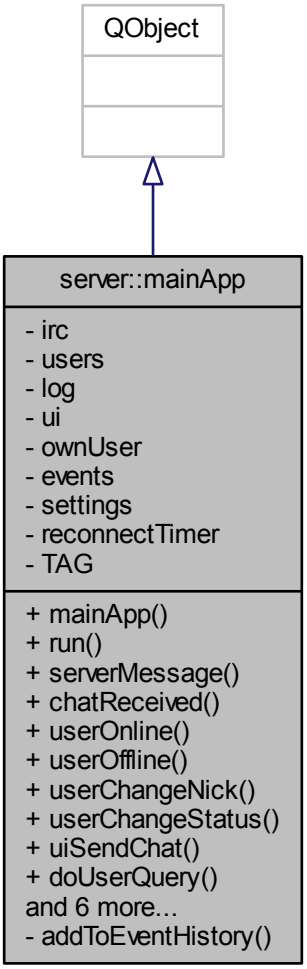
- C++/resources/jsoncommand.h
- C++/resources/jsoncommand.cpp

## 7.28 server::mainApp Class Reference

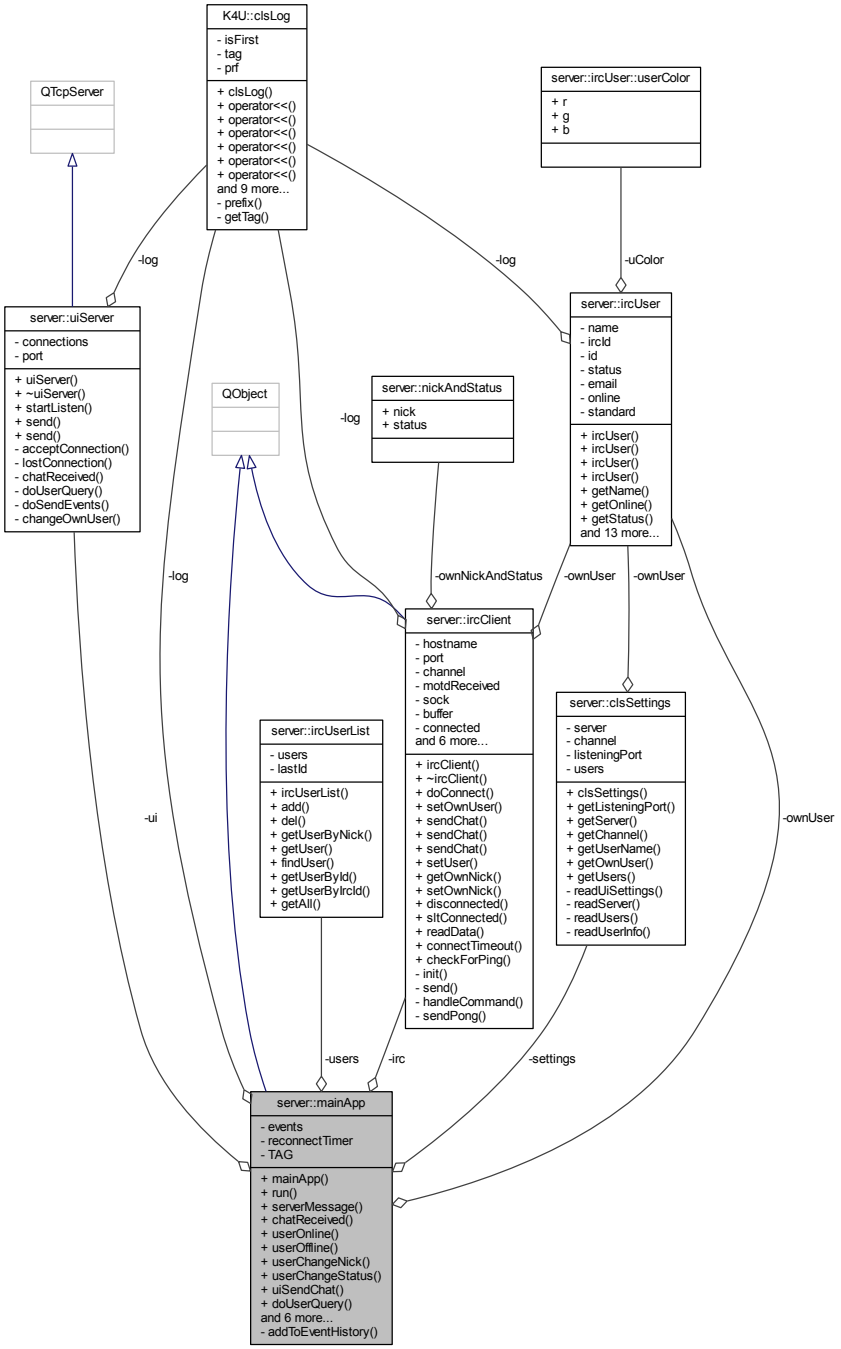
The main application.

```
#include <mainapp.h>
```

Inheritance diagram for server::mainApp:



Collaboration diagram for server::mainApp:




## Public Slots

- void chatReceived (const QString channel, const nickAndStatus user, const QString message)

  *Triggered when a chat has been received.*
- void userOnline (const nickAndStatus nick, const QString ircId, const QString channel)

  *Triggered when a user joins trough a JOIN message.*
- void userOffline (const nickAndStatus nick, const QString id, const QString channel)

  *Triggered when a user leaves trough a PART message.*

- void userChangeNick (const nickAndStatus oldNick, const nickAndStatus newNick, const QString id)

  *Triggered when a user changes his nickname (nick|Status)*
- void userChangeStatus (const nickAndStatus oldNick, const nickAndStatus newNick, const QString id)

  *Triggered when a user changes his status (nick|Status)*
- void uiSendChat (const QString message)

  *Triggered when a UI client wants to send a chat message.*
- void doUserQuery (uiClient *client)

  *Triggered when a UI client wants all the users.*
- void doSendEvents (uiClient *client)

  *Triggered when a UI client wants all the events known to the server.*
- void ownUserChangeStatus (QString newStatus)

  *Triggered when a UI client wants to change the status.*
- void ircConnectTimeOut ()

  *Triggered when the IRC client cannot connect.*
- void ircConnected ()

  *Triggered when the IRC client has connected.*
- void ircDisconnected (QString reason)

  *Triggered when the IRC client has disconnected.*
- void doIrcReconnect ()

  *Emitted when the reconnect timer reaches 0.*

## Public Member Functions

- mainApp (QString parentPath, QObject *parent=0)
- void run ()
- void serverMessage (const QString &message, const level lvl)

  *For logging to the clients.*

## Private Member Functions

- void addToEventHistory (clsEvent *event)

  *Adds an event to the history, checking for the maximum number.*

## Private Attributes

- ircClient * irc
- ircUserList * users
- clsLog log
- uiServer * ui
- ircUser * ownUser
- QVector< clsEvent * > events
- clsSettings * settings
- QTimer reconnectTimer

## Static Private Attributes

- static const logTags TAG = LOGTAGS_MAIN

### 7.28.1 Detailed Description

The main application.

**Author**

Koen Beckers (K-4U)

Definition at line 25 of file mainapp.h.

### 7.28.2 Constructor & Destructor Documentation

#### 7.28.2.1 server::mainApp::mainApp ( QString *parentPath*, QObject * *parent* = 0 ) [explicit]

Definition at line 5 of file mainapp.cpp.

### 7.28.3 Member Function Documentation

#### 7.28.3.1 void server::mainApp::addToEventHistory ( clsEvent * *event* ) [private]

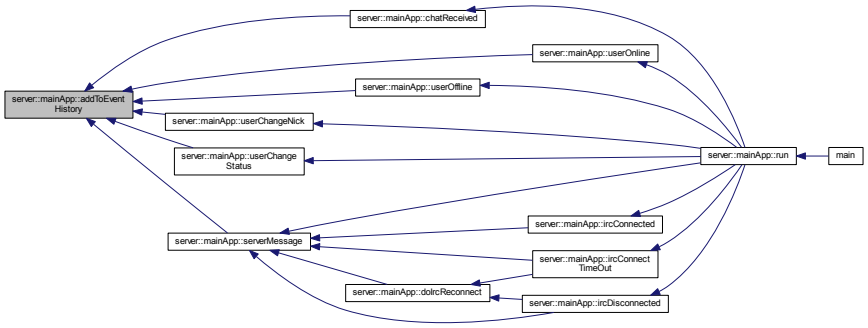
Adds an event to the history, checking for the maximum number.

**Parameters**

| | |
|---|---|
| *event* | The event to add The event can be any event, as these classes are derived from eachother |

Definition at line 67 of file mainapp.cpp.

Here is the caller graph for this function:




#### 7.28.3.2 void server::mainApp::chatReceived ( const QString *channel*, const nickAndStatus *user*, const QString *message* ) [slot]

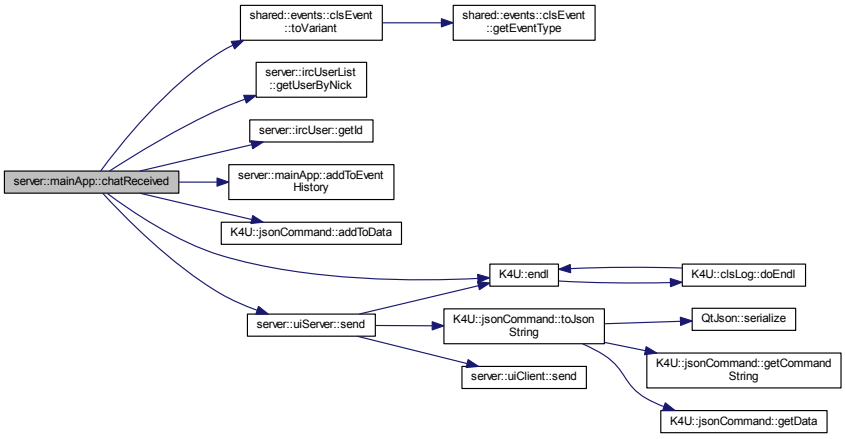
Triggered when a chat has been received.

**Parameters**

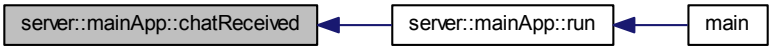
| | |
|---|---|
| *channel* | The channel in which the chat has been received |
| *user* | The nick and status of the user that send the chat |
| *message* | The message |

Definition at line 74 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

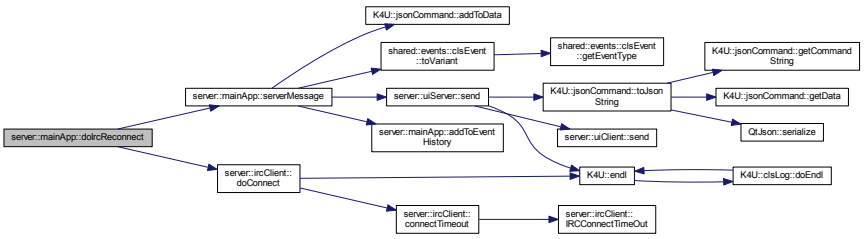



#### 7.28.3.3 void server::mainApp::doIrcReconnect ( ) [slot]

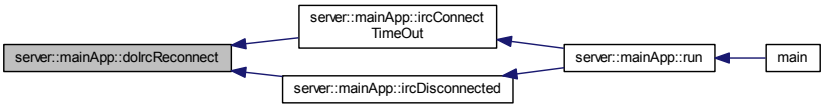
Emitted when the reconnect timer reaches 0.

Definition at line 264 of file mainapp.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:




#### 7.28.3.4 void server::mainApp::doSendEvents ( uiClient * *client* ) [slot]

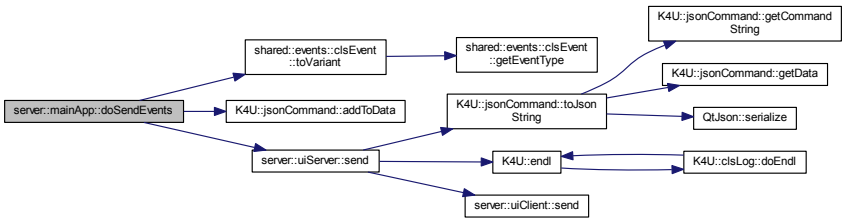
Triggered when a UI client wants all the events known to the server.

**Parameters**

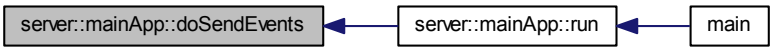
| | |
|---|---|
| *client* | The client which requested the events |

Definition at line 203 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.28.3.5 void server::mainApp::doUserQuery ( uiClient * *client* ) [slot]

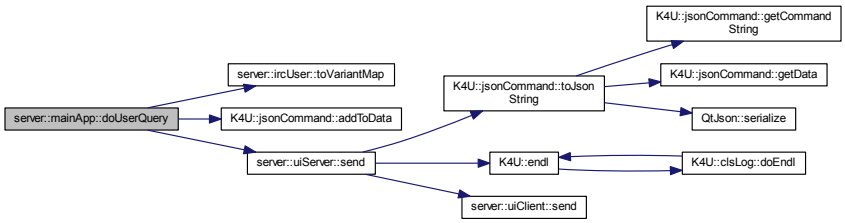
Triggered when a UI client wants all the users.

**Parameters**

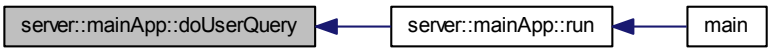
| | |
|---|---|
| *client* | The client which requested the user query |

Definition at line 190 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

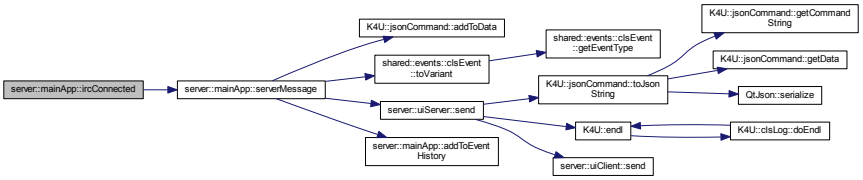



#### 7.28.3.6 void server::mainApp::ircConnected ( ) [slot]

Triggered when the IRC client has connected.

Definition at line 250 of file mainapp.cpp.

Here is the call graph for this function:



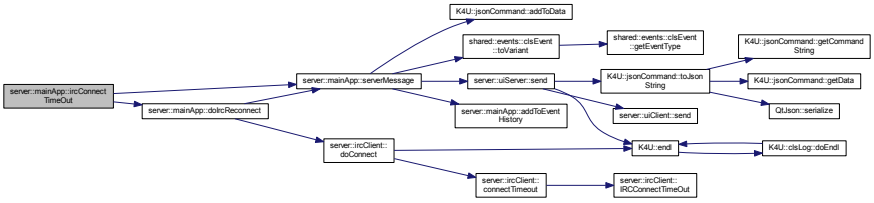

Here is the caller graph for this function:

#### 7.28.3.7 void server::mainApp::ircConnectTimeOut ( ) [slot]

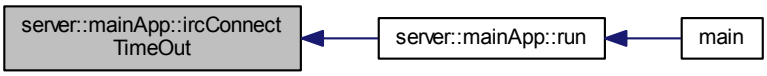
Triggered when the IRC client cannot connect.

Definition at line 240 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

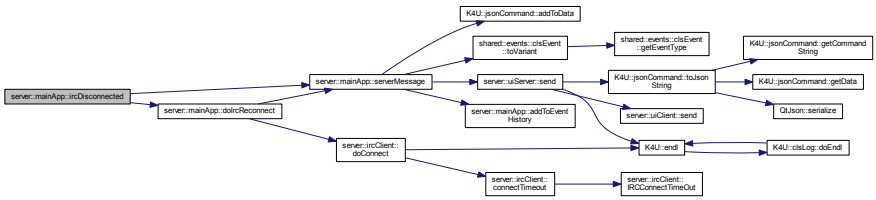



#### 7.28.3.8 void server::mainApp::ircDisconnected ( QString *reason* ) [slot]

Triggered when the IRC client has disconnected.

Definition at line 254 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.28.3.9 void server::mainApp::ownUserChangeStatus ( QString *newStatus* ) [slot]

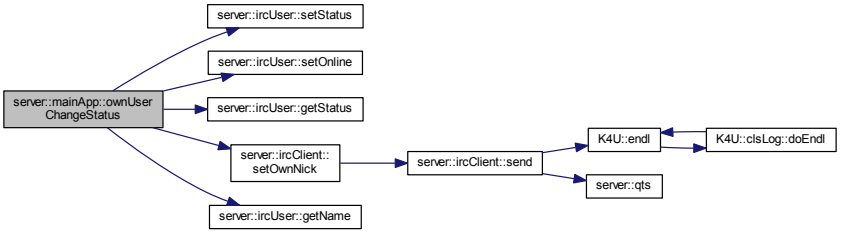
Triggered when a UI client wants to change the status.

**Parameters**

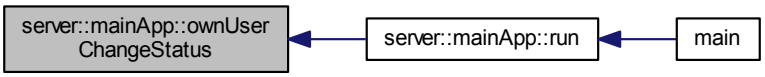
| | |
|---|---|
| *newStatus* | The status to change to |

Definition at line 214 of file mainapp.cpp.

Here is the call graph for this function:



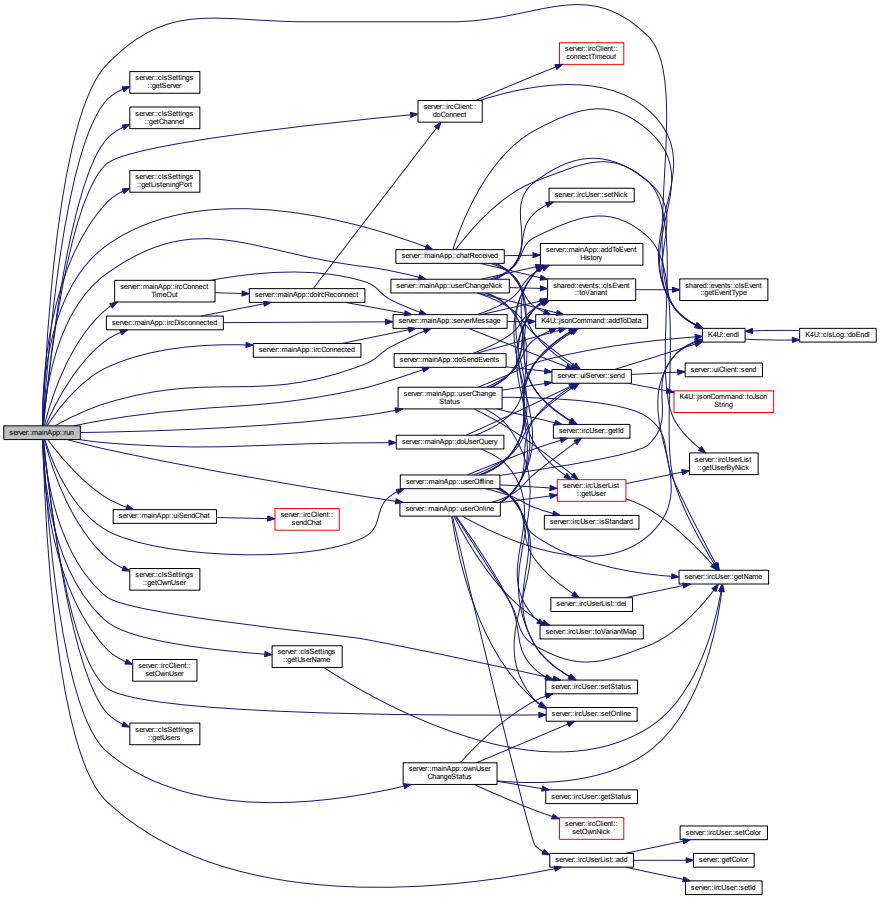

Here is the caller graph for this function:




#### 7.28.3.10 void server::mainApp::run ( )

Definition at line 14 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.28.3.11 void server::mainApp::serverMessage ( const QString & *message*, const level *lvl* )

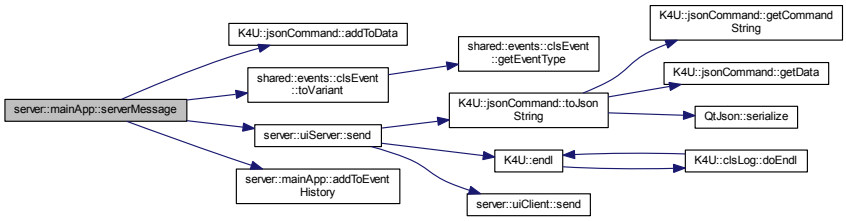
For logging to the clients.

**Parameters**

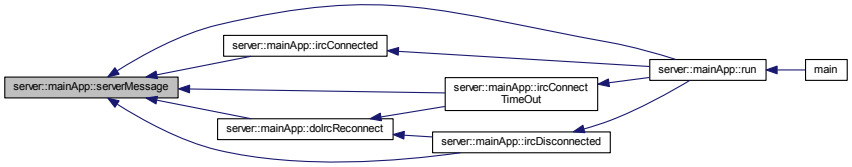
| | |
|---|---|
| *message* | The message to log |

Definition at line 229 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.28.3.12 void server::mainApp::uiSendChat ( const QString *message* ) [slot]

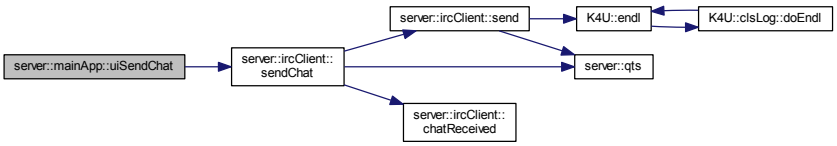
Triggered when a UI client wants to send a chat message.

**Parameters**

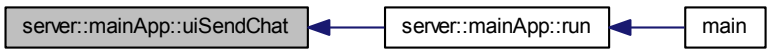
| | |
|---|---|
| *message* | The message to send |

Definition at line 186 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:



#### 7.28.3.13 void server::mainApp::userChangeNick ( const nickAndStatus *oldNick*, const nickAndStatus *newNick*, const QString *id* ) [slot]

Triggered when a user changes his nickname (nick|Status)

**Parameters**

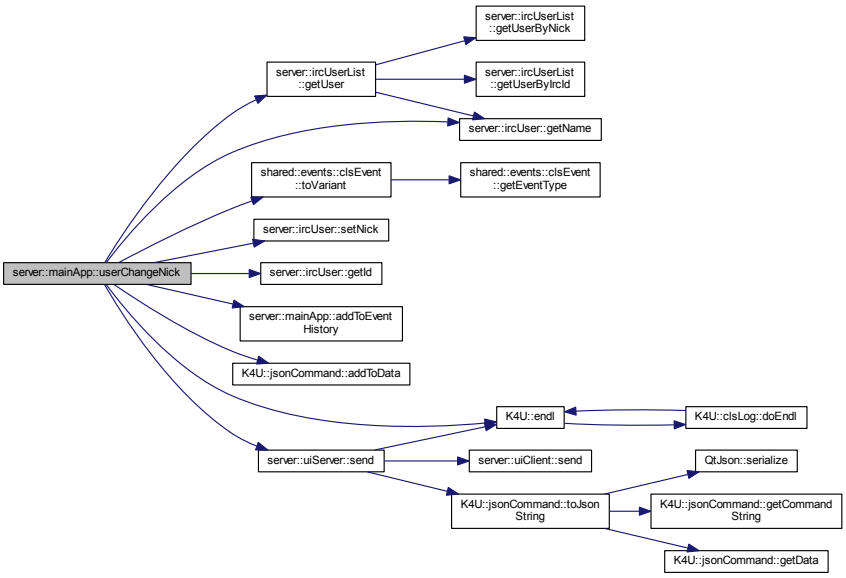
| | |
|---|---|
| *oldNickName* | The old nick and status this user had |
| *newNickName* | The new nick and status this user now has |
| *id* | His IRC-ID. The only thing probably that remained the same. |

**Warning**

Not to be confused with userChangeStatus!

Definition at line 138 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.28.3.14 void server::mainApp::userChangeStatus ( const nickAndStatus *oldNick*, const nickAndStatus *newNick*, const QString *id* ) [slot]

Triggered when a user changes his status (nick|Status)

**Parameters**

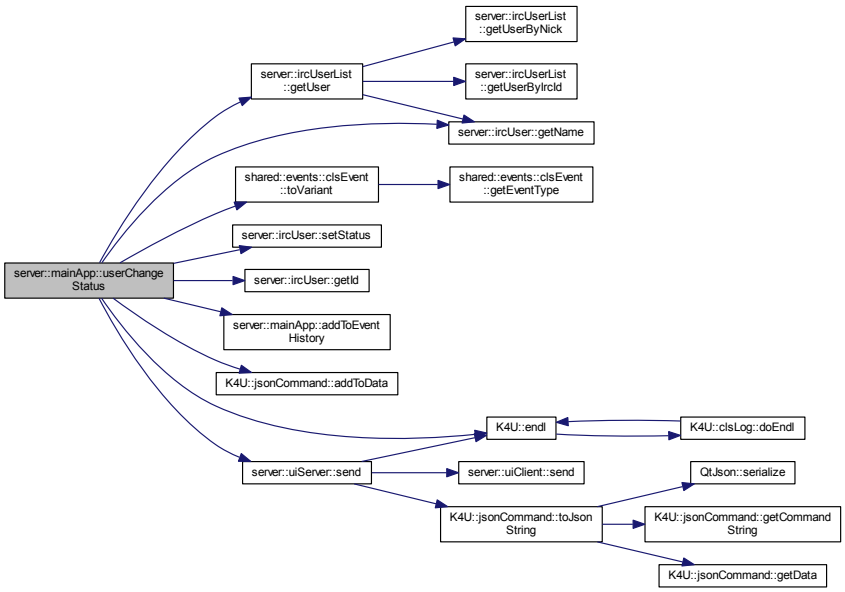
| | |
|---|---|
| *oldNick* | The old nick and status this user had |
| *newNick* | The new nick and status this user now has |
| *id* | His IRC-ID. The only thing probably that remained the same. |

**Warning**

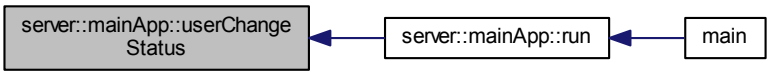
Not to be confused with userChangeNick!

Definition at line 162 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.28.3.15 void server::mainApp::userOffline ( const nickAndStatus *nick*, const QString *id*, const QString *channel* ) [slot]

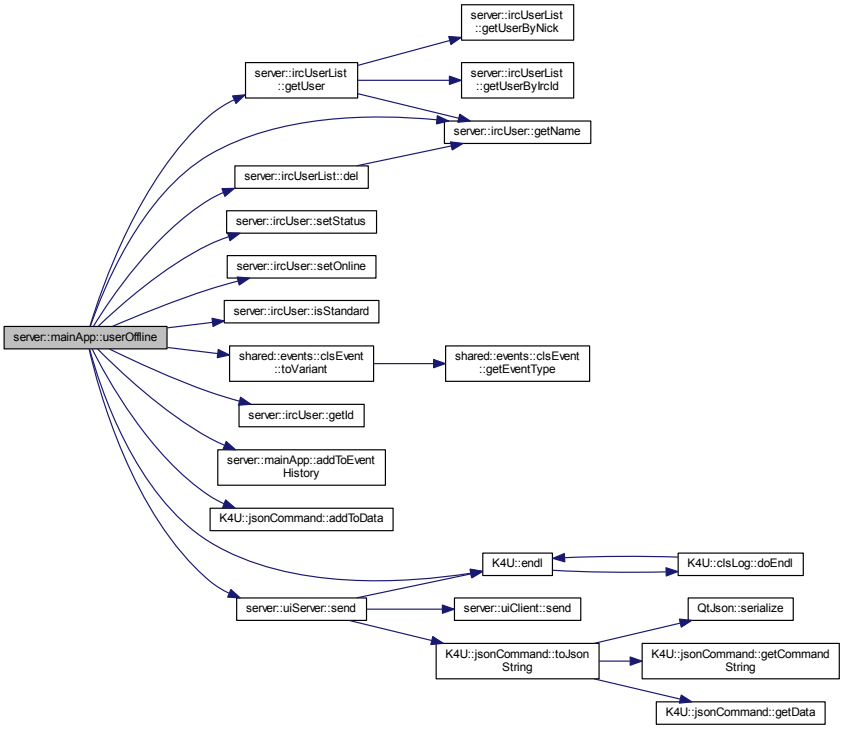
Triggered when a user leaves trough a PART message.

**Parameters**

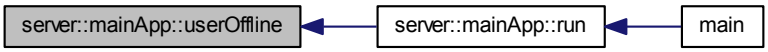
| | |
|---|---|
| *nick* | The nick and status of the user that left |
| *id* | The IRC-ID of the user that left |
| *channel* | The channel which the user has left |

Definition at line 113 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.28.3.16 void server::mainApp::userOnline ( const **nickAndStatus** *nick,* const QString *ircId,* const QString *channel* ) [slot]

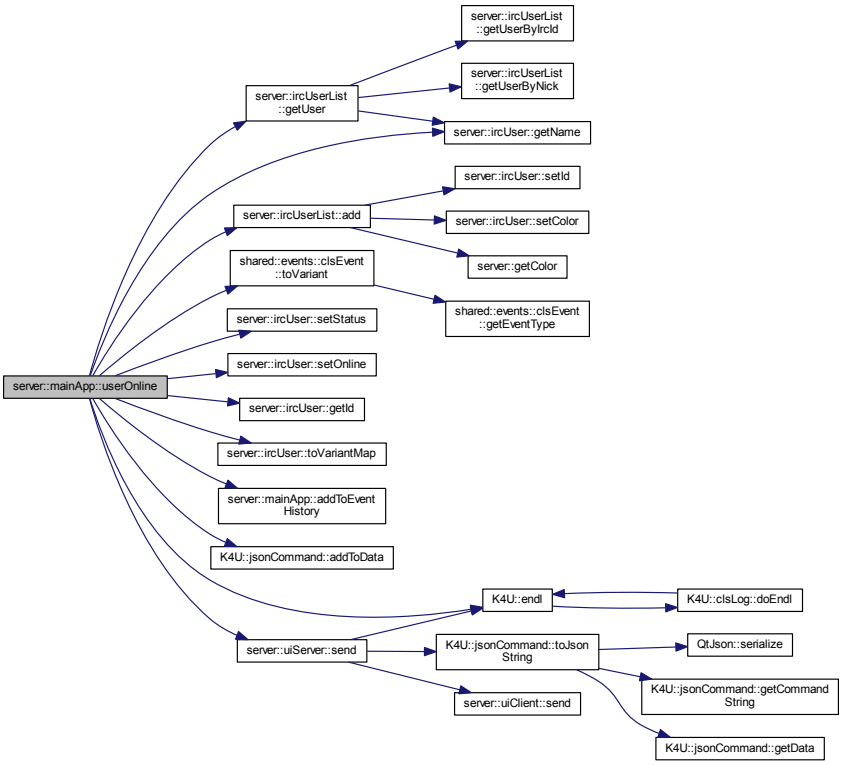
Triggered when a user joins trough a JOIN message.

**Parameters**

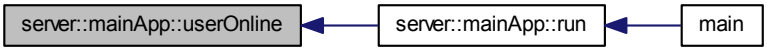
| | |
|---|---|
| *nick* | The nick and status of the user that joins |
| *ircId* | The IRC-ID of the user that joins |
| *channel* | The channel in which the user has joined |

Definition at line 92 of file mainapp.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




## 7.28.4 Member Data Documentation

### 7.28.4.1 QVector<clsEvent *> server::mainApp::events [private]

Definition at line 146 of file mainapp.h.

### 7.28.4.2 ircClient* server::mainApp::irc [private]

Definition at line 140 of file mainapp.h.

### 7.28.4.3 clsLog server::mainApp::log [private]

Definition at line 142 of file mainapp.h.

#### 7.28.4.4 ircUser* server::mainApp::ownUser [private]

Definition at line 145 of file mainapp.h.

#### 7.28.4.5 QTimer server::mainApp::reconnectTimer [private]

Definition at line 150 of file mainapp.h.

#### 7.28.4.6 clsSettings* server::mainApp::settings [private]

Definition at line 148 of file mainapp.h.

#### 7.28.4.7 const logTags server::mainApp::TAG = LOGTAGS_MAIN [static], [private]

Definition at line 139 of file mainapp.h.

#### 7.28.4.8 uiServer* server::mainApp::ui [private]

Definition at line 143 of file mainapp.h.

#### 7.28.4.9 ircUserList* server::mainApp::users [private]

Definition at line 141 of file mainapp.h.

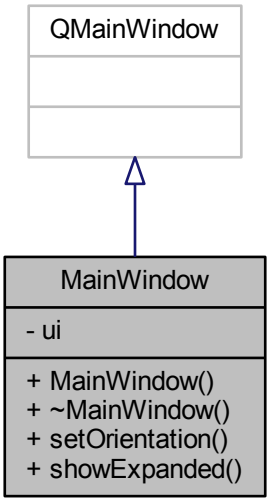
The documentation for this class was generated from the following files:

- C++/customIrcServer/mainapp.h
- C++/customIrcServer/mainapp.cpp

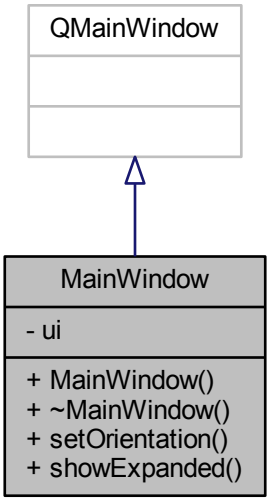
## 7.29 MainWindow Class Reference

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

Inheritance diagram for MainWindow:




Collaboration diagram for MainWindow:




## Public Types

- enum ScreenOrientation { ScreenOrientationLockPortrait, ScreenOrientationLockLandscape, ScreenOrientationAuto }

### Public Member Functions

- MainWindow (QWidget *parent=0)
- virtual ~MainWindow ()
- void setOrientation (ScreenOrientation orientation)
- void showExpanded ()

### Private Attributes

- Ui::MainWindow * ui

## 7.29.1 Detailed Description

Definition at line 10 of file mainwindow.h.

## 7.29.2 Member Enumeration Documentation

### 7.29.2.1 enum MainWindow::ScreenOrientation

**Enumerator**

***ScreenOrientationLockPortrait***

***ScreenOrientationLockLandscape***

***ScreenOrientationAuto***

Definition at line 14 of file mainwindow.h.

## 7.29.3 Constructor & Destructor Documentation

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

Definition at line 6 of file mainwindow.cpp.

### 7.29.3.2 MainWindow::~MainWindow ( ) [virtual]

Definition at line 12 of file mainwindow.cpp.

## 7.29.4 Member Function Documentation

### 7.29.4.1 void MainWindow::setOrientation ( ScreenOrientation *orientation* )

Definition at line 17 of file mainwindow.cpp.

Here is the caller graph for this function:

#### 7.29.4.2 void MainWindow::showExpanded ( )

Definition at line 49 of file mainwindow.cpp.

Here is the caller graph for this function:

### 7.29.5 Member Data Documentation

#### 7.29.5.1 Ui::MainWindow* MainWindow::ui [private]

Definition at line 29 of file mainwindow.h.

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

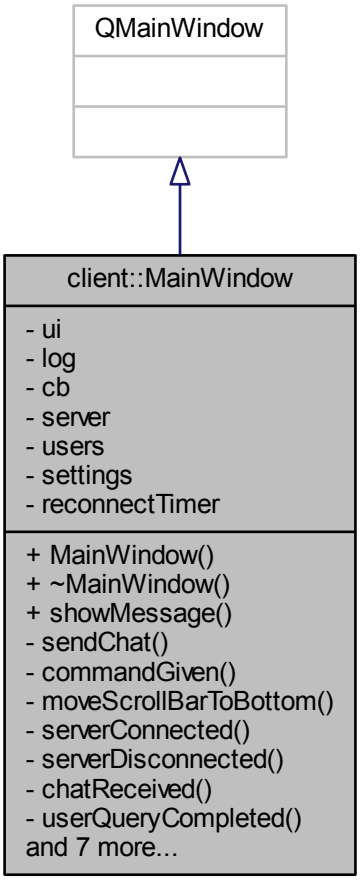
- C++/customIrcClient/android/mainwindow.h
- C++/customIrcClient/android/mainwindow.cpp

## 7.30 client::MainWindow Class Reference

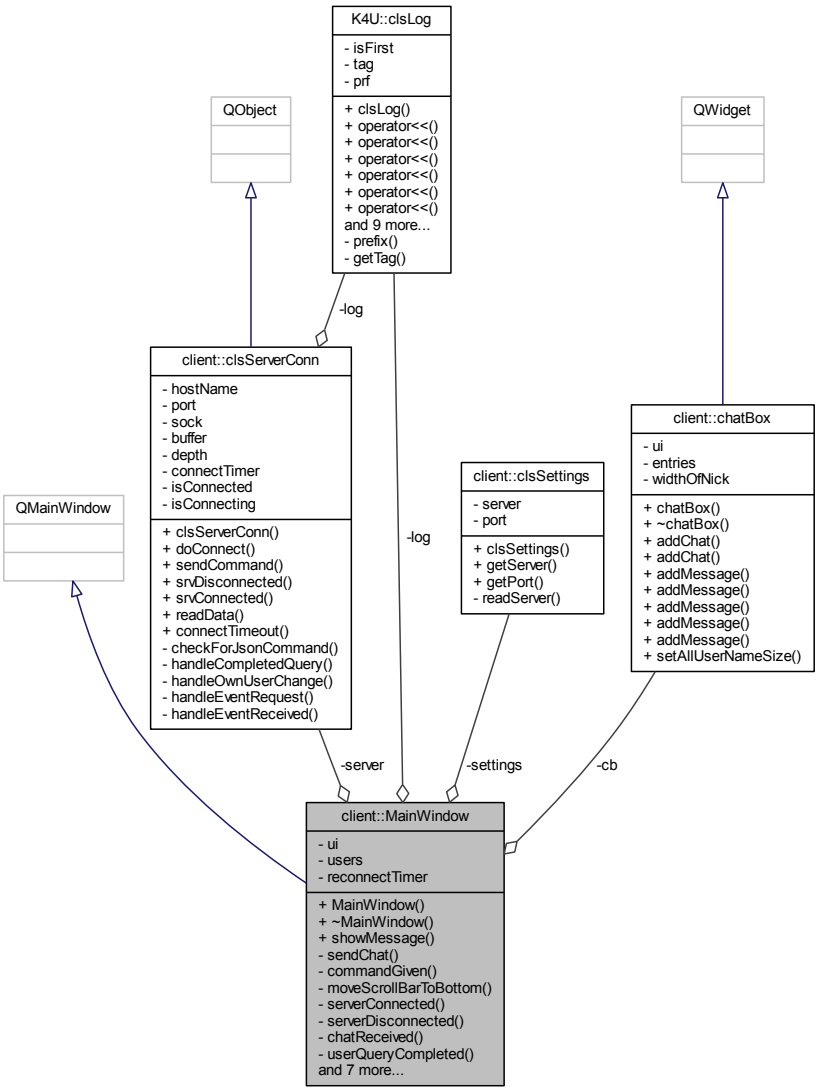
Main UI window.

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

Inheritance diagram for client::MainWindow:



Collaboration diagram for client::MainWindow:




## Public Member Functions

- MainWindow (QApplication ∗a, QWidget ∗parent=0)
- ∼MainWindow ()
- void showMessage (QString msg, level lvl)

## Private Slots

- void sendChat (QString message)
- void commandGiven (QString command, QList< QString > args)
- void moveScrollBarToBottom (int min, int max)
- void serverConnected ()
- void serverDisconnected ()
- void chatReceived (eventChat ∗chatEvent)

*Triggered when a chat has been received.*

- void userQueryCompleted (QVector< ircUser * > users)

  *Triggered when the user query came back.*

- void userStatusChange (eventUserChangeStatus *statusEvent)

  *Triggered when a user changes his status (nick|status)*

- void userChangeNick (eventUserChangeNick *nickEvent)

  *Triggered when a user changes his nick (nick|status)*

- void userLeave (eventUserLeave *leaveEvent)

  *Triggered when a user leaves the room.*

- void userEnter (eventUserJoin *newUser)

  *Triggered when a user enters the room.*

- void serverMessageReceived (eventServerMessage *myMessage)

  *Triggered when the server sends us a message that should be displayed in the UI.*

- void connectedTimeout ()

  *Triggered when we cannot connect to the server!*

- void doReconnect ()

### Private Attributes

- Ui::MainWindow * ui
- clsLog log
- chatBox * cb
- clsServerConn * server
- QMap< int, ircUser * > users
- clsSettings * settings
- QTimer * reconnectTimer

## 7.30.1 Detailed Description

Main UI window.

**Author**

Koen Beckers (K-4U) This main window is the same for android and desktop Please mind the defines

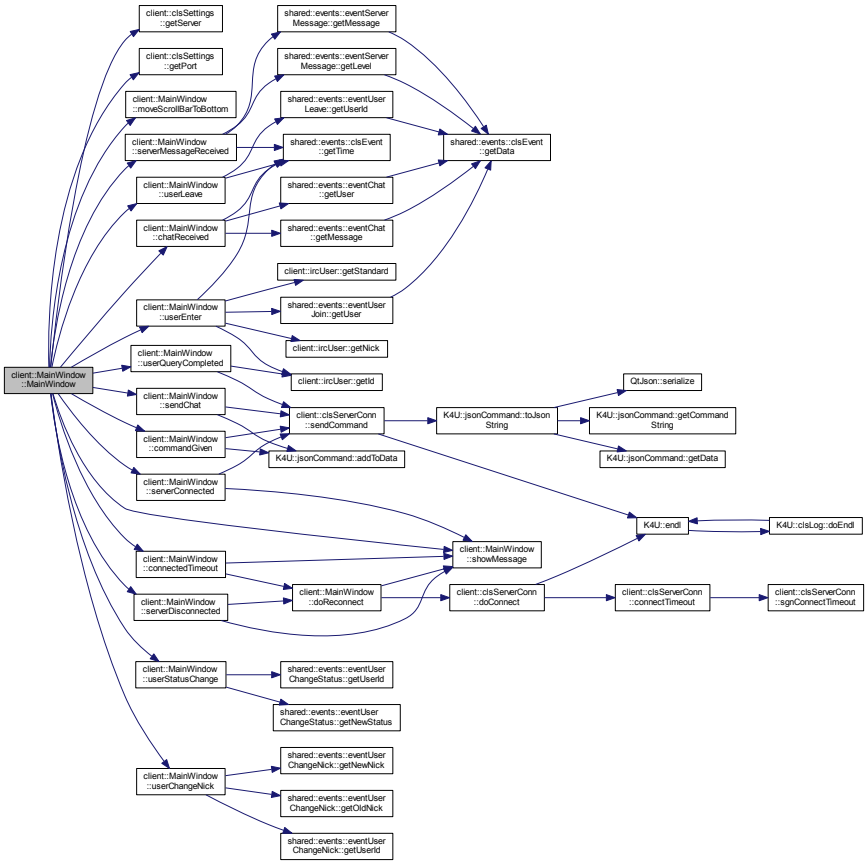
Definition at line 33 of file mainwindow.h.

## 7.30.2 Constructor & Destructor Documentation

### 7.30.2.1 MainWindow::MainWindow ( QApplication * *a*, QWidget * *parent* = 0 ) [explicit]

Definition at line 5 of file mainwindow.cpp.

Here is the call graph for this function:




#### 7.30.2.2 MainWindow::~MainWindow ( )

Definition at line 54 of file mainwindow.cpp.

### 7.30.3 Member Function Documentation

#### 7.30.3.1 void MainWindow::chatReceived ( eventChat * *chatEvent* ) [private], [slot]

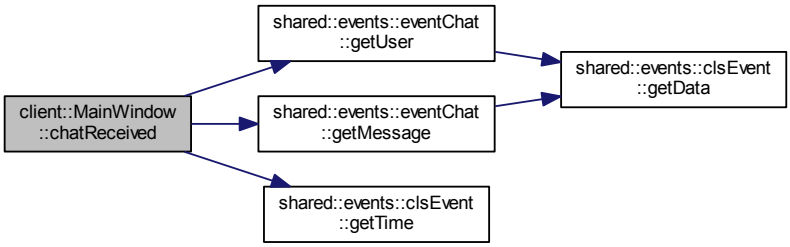
Triggered when a chat has been received.

**Parameters**

| | |
|---|---|
| *chatEvent* | An eventChat object containing more data |

Definition at line 60 of file mainwindow.cpp.

Here is the call graph for this function:

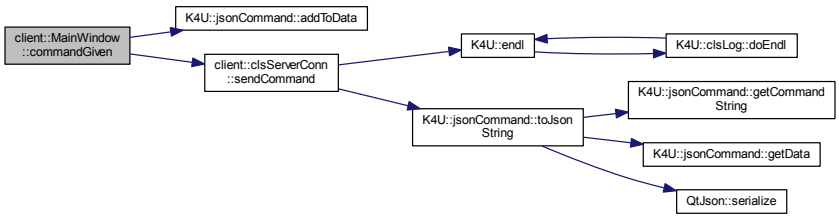



Here is the caller graph for this function:

#### 7.30.3.2 void MainWindow::commandGiven ( QString *command,* QList< QString > *args* ) [private],[slot]

Definition at line 72 of file mainwindow.cpp.

Here is the call graph for this function:



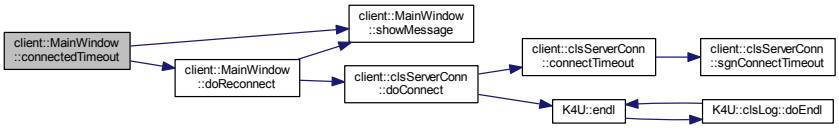
Here is the caller graph for this function:

#### 7.30.3.3 void MainWindow::connectedTimeout ( ) [private],[slot]

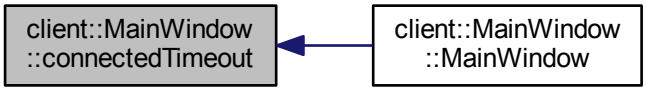
Triggered when we cannot connect to the server!

Definition at line 226 of file mainwindow.cpp.

Here is the call graph for this function:



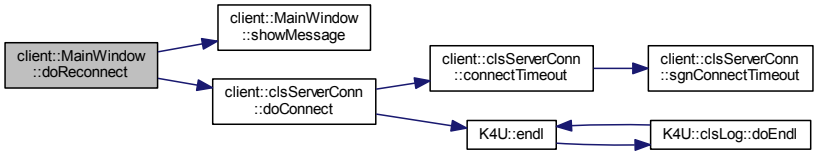

Here is the caller graph for this function:



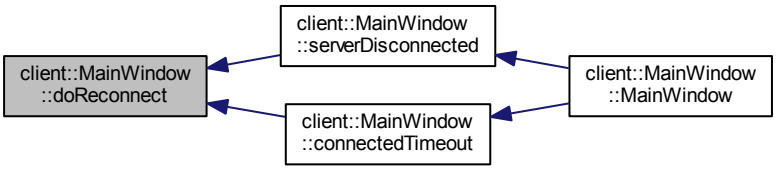

#### 7.30.3.4 void MainWindow::doReconnect ( ) [private],[slot]

Definition at line 235 of file mainwindow.cpp.

Here is the call graph for this function:



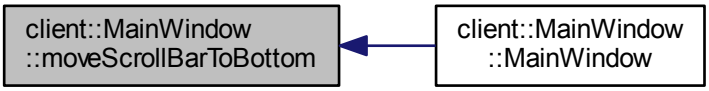

Here is the caller graph for this function:




#### 7.30.3.5 void MainWindow::moveScrollBarToBottom ( int *min*, int *max* ) [private],[slot]

Definition at line 86 of file mainwindow.cpp.

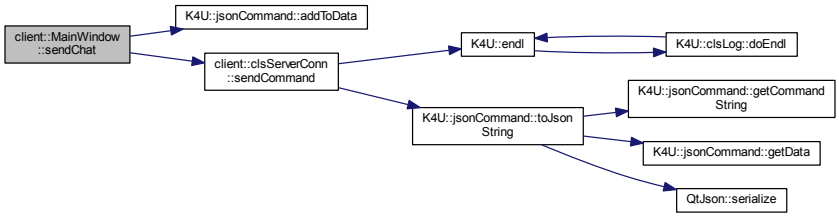
Here is the caller graph for this function:




#### 7.30.3.6 void MainWindow::sendChat ( QString *message* ) [private],[slot]

Definition at line 65 of file mainwindow.cpp.

Here is the call graph for this function:

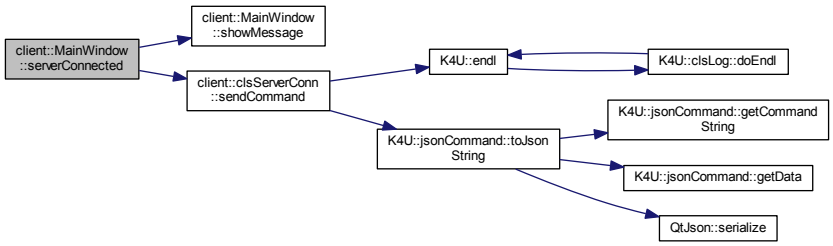



Here is the caller graph for this function:

#### 7.30.3.7 void MainWindow::serverConnected ( ) [private],[slot]

Definition at line 91 of file mainwindow.cpp.

Here is the call graph for this function:

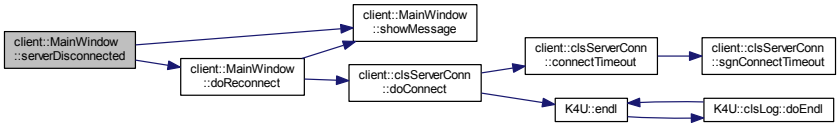


Here is the caller graph for this function:

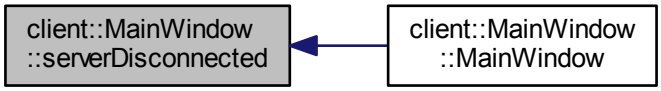
#### 7.30.3.8 void MainWindow::serverDisconnected ( ) [private],[slot]

Definition at line 100 of file mainwindow.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.30.3.9 void MainWindow::serverMessageReceived ( eventServerMessage * *myMessage* ) [private],[slot]

Triggered when the server sends us a message that should be displayed in the UI.

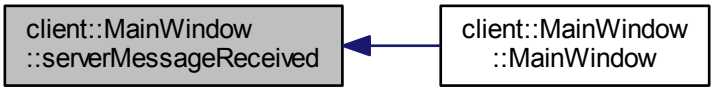
**Parameters**

| | |
|---|---|
| *myMessage* | An eventServerMessage object containing more data |

Definition at line 195 of file mainwindow.cpp.

Here is the call graph for this function:

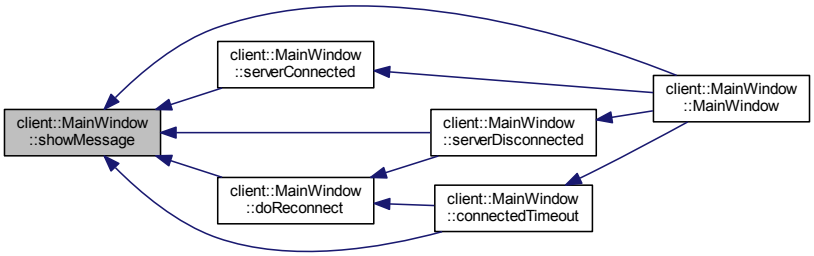
Here is the caller graph for this function:




#### 7.30.3.10 void MainWindow::showMessage ( QString *msg*, level *lvl* )

Definition at line 212 of file mainwindow.cpp.

Here is the caller graph for this function:




#### 7.30.3.11 void MainWindow::userChangeNick ( eventUserChangeNick * *nickEvent* ) [private],[slot]

Triggered when a user changes his nick (nick|status)

**Parameters**

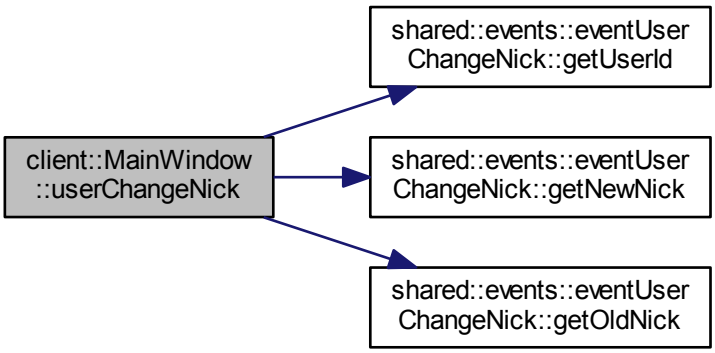
| | |
|---|---|
| *nickEvent* | An eventUserChangeNick object containing more data |

**Note**

Not to be confused with userStatusChange

Definition at line 139 of file mainwindow.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.30.3.12 void MainWindow::userEnter ( eventUserJoin * *newUser* ) [private],[slot]

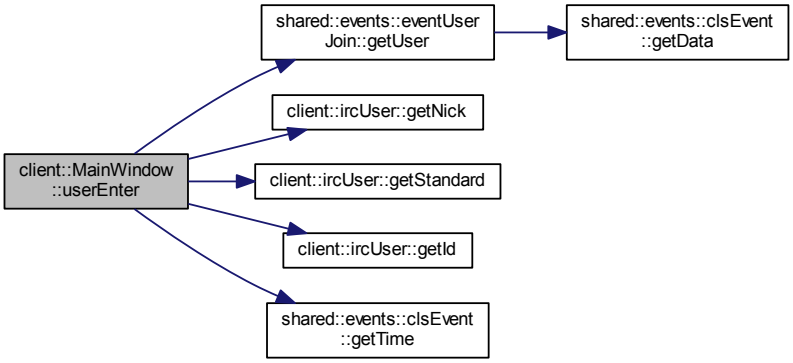
Triggered when a user enters the room.

**Parameters**

| | |
|---|---|
| *newUser* | An eventUserJoin object containing more data |

Definition at line 172 of file mainwindow.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.30.3.13 void MainWindow::userLeave ( eventUserLeave * *leaveEvent* ) [private], [slot]

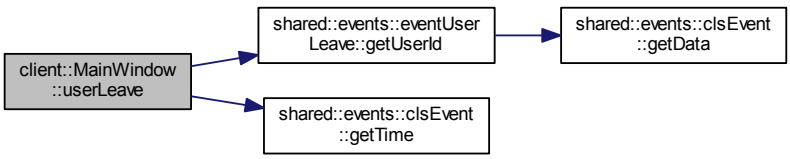
Triggered when a user leaves the room.

**Parameters**

| | |
|---|---|
| *leaveEvent* | An eventUserLeave object containing more data |

Definition at line 149 of file mainwindow.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

#### 7.30.3.14 void MainWindow::userQueryCompleted ( QVector< ircUser * > *users* ) [private],[slot]

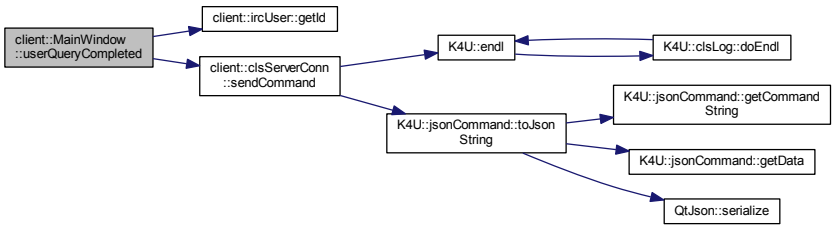
Triggered when the user query came back.

**Parameters**

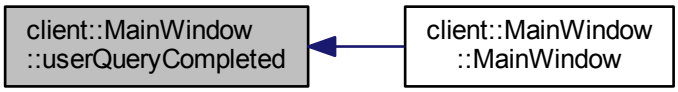
| | |
|---|---|
| *users* | A QVector<ircUser*> that contains all the users online at that point |

Definition at line 109 of file mainwindow.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.30.3.15 void MainWindow::userStatusChange ( eventUserChangeStatus * *statusEvent* ) [private],[slot]

Triggered when a user changes his status (nick|status)

**Parameters**

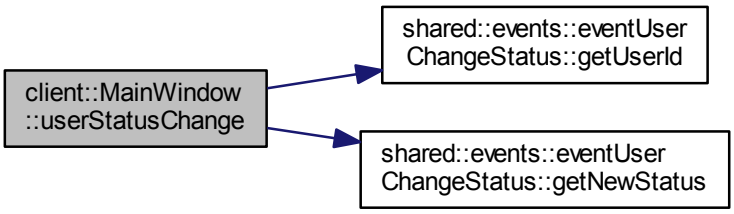
| | |
|---|---|
| *statusEvent* | An eventUserChangeStatus object containing more data |

**Note**

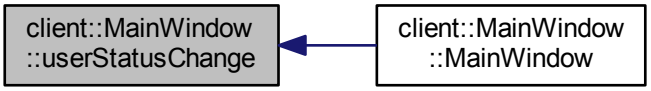
Not to be confused with userChangeNick!

Definition at line 129 of file mainwindow.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




### 7.30.4 Member Data Documentation

#### 7.30.4.1 chatBox* client::MainWindow::cb [private]

Definition at line 115 of file mainwindow.h.

#### 7.30.4.2 clsLog client::MainWindow::log [private]

Definition at line 114 of file mainwindow.h.

#### 7.30.4.3 QTimer* client::MainWindow::reconnectTimer [private]

Definition at line 122 of file mainwindow.h.

#### 7.30.4.4 clsServerConn* client::MainWindow::server [private]

Definition at line 117 of file mainwindow.h.

#### 7.30.4.5 clsSettings* client::MainWindow::settings [private]

Definition at line 121 of file mainwindow.h.

#### 7.30.4.6 Ui::MainWindow* client::MainWindow::ui [private]

Definition at line 112 of file mainwindow.h.

#### 7.30.4.7 QMap<int, ircUser *> client::MainWindow::users [private]

Definition at line 119 of file mainwindow.h.

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

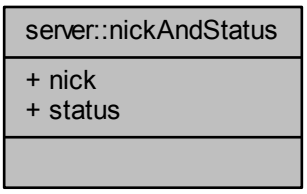
- C++/customIrcClient/mainwindow.h
- C++/customIrcClient/mainwindow.cpp

## 7.31 server::nickAndStatus Struct Reference

The nickAndStatus struct.

`#include <functions.h>`

Collaboration diagram for server::nickAndStatus:




### Public Attributes

- QString nick
- QString status

### 7.31.1 Detailed Description

The nickAndStatus struct.

Definition at line 13 of file functions.h.

### 7.31.2 Member Data Documentation

#### 7.31.2.1 QString server::nickAndStatus::nick

Definition at line 14 of file functions.h.

#### 7.31.2.2 QString server::nickAndStatus::status

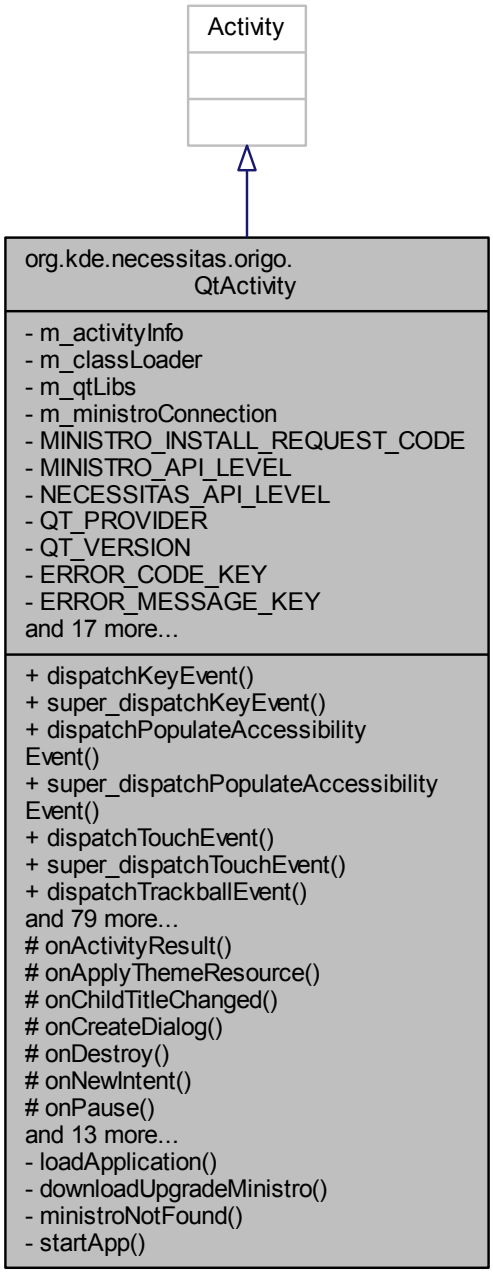
Definition at line 15 of file functions.h.

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

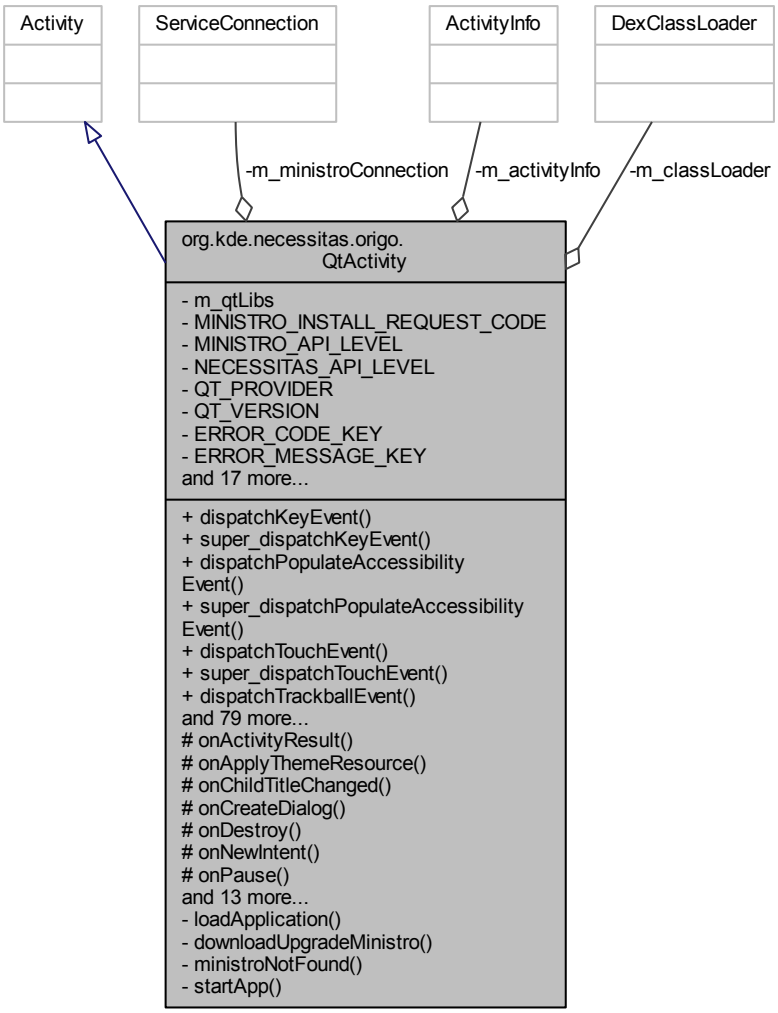
- C++/customIrcServer/functions.h

## 7.32 org.kde.necessitas.origo.QtActivity Class Reference

Inheritance diagram for org.kde.necessitas.origo.QtActivity:



Collaboration diagram for org.kde.necessitas.origo.QtActivity:




## Public Member Functions

- boolean dispatchKeyEvent (KeyEvent event)
- boolean super_dispatchKeyEvent (KeyEvent event)
- boolean dispatchPopulateAccessibilityEvent (AccessibilityEvent event)
- boolean super_dispatchPopulateAccessibilityEvent (AccessibilityEvent event)
- boolean dispatchTouchEvent (MotionEvent ev)
- boolean super_dispatchTouchEvent (MotionEvent event)
- boolean dispatchTrackballEvent (MotionEvent ev)
- boolean super_dispatchTrackballEvent (MotionEvent event)
- void super_onActivityResult (int requestCode, int resultCode, Intent data)
- void super_onApplyThemeResource (Theme theme, int resid, boolean first)
- void super_onChildTitleChanged (Activity childActivity, CharSequence title)
- void onConfigurationChanged (Configuration newConfig)
- void super_onConfigurationChanged (Configuration newConfig)
- void onContentChanged ()

- void super_onContentChanged ()
- boolean onContextItemSelected (MenuItem item)
- boolean super_onContextItemSelected (MenuItem item)
- void onContextMenuClosed (Menu menu)
- void super_onContextMenuClosed (Menu menu)
- void onCreate (Bundle savedInstanceState)
- void onCreateContextMenu (ContextMenu menu, View v, ContextMenuInfo menuInfo)
- void super_onCreateContextMenu (ContextMenu menu, View v, ContextMenuInfo menuInfo)
- CharSequence onCreateDescription ()
- CharSequence super_onCreateDescription ()
- Dialog super_onCreateDialog (int id)
- boolean onCreateOptionsMenu (Menu menu)
- boolean super_onCreateOptionsMenu (Menu menu)
- boolean onCreatePanelMenu (int featureId, Menu menu)
- boolean super_onCreatePanelMenu (int featureId, Menu menu)
- View onCreatePanelView (int featureId)
- View super_onCreatePanelView (int featureId)
- boolean onCreateThumbnail (Bitmap outBitmap, Canvas canvas)
- boolean super_onCreateThumbnail (Bitmap outBitmap, Canvas canvas)
- View onCreateView (String name, Context context, AttributeSet attrs)
- View super_onCreateView (String name, Context context, AttributeSet attrs)
- boolean onKeyDown (int keyCode, KeyEvent event)
- boolean super_onKeyDown (int keyCode, KeyEvent event)
- boolean onKeyMultiple (int keyCode, int repeatCount, KeyEvent event)
- boolean super_onKeyMultiple (int keyCode, int repeatCount, KeyEvent event)
- boolean onKeyUp (int keyCode, KeyEvent event)
- boolean super_onKeyUp (int keyCode, KeyEvent event)
- void onLowMemory ()
- boolean onMenuItemSelected (int featureId, MenuItem item)
- boolean super_onMenuItemSelected (int featureId, MenuItem item)
- boolean onMenuOpened (int featureId, Menu menu)
- boolean super_onMenuOpened (int featureId, Menu menu)
- void super_onNewIntent (Intent intent)
- boolean onOptionsItemSelected (MenuItem item)
- boolean super_onOptionsItemSelected (MenuItem item)
- void onOptionsMenuClosed (Menu menu)
- void super_onOptionsMenuClosed (Menu menu)
- void onPanelClosed (int featureId, Menu menu)
- void super_onPanelClosed (int featureId, Menu menu)
- void super_onPrepareDialog (int id, Dialog dialog)
- boolean onPrepareOptionsMenu (Menu menu)
- boolean super_onPrepareOptionsMenu (Menu menu)
- boolean onPreparePanel (int featureId, View view, Menu menu)
- boolean super_onPreparePanel (int featureId, View view, Menu menu)
- void super_onRestoreInstanceState (Bundle savedInstanceState)
- Object onRetainNonConfigurationInstance ()
- Object super_onRetainNonConfigurationInstance ()
- void super_onSaveInstanceState (Bundle outState)
- boolean onSearchRequested ()
- boolean super_onSearchRequested ()
- void super_onTitleChanged (CharSequence title, int color)
- boolean onTouchEvent (MotionEvent event)
- boolean super_onTouchEvent (MotionEvent event)
- boolean onTrackballEvent (MotionEvent event)
- boolean super_onTrackballEvent (MotionEvent event)

- void onUserInteraction ()
- void super_onUserInteraction ()
- void super_onUserLeaveHint ()
- void onWindowAttributesChanged (LayoutParams params)
- void super_onWindowAttributesChanged (LayoutParams params)
- void onWindowFocusChanged (boolean hasFocus)
- void super_onWindowFocusChanged (boolean hasFocus)
- void onAttachedToWindow ()
- void super_onAttachedToWindow ()
- void onBackPressed ()
- void super_onBackPressed ()
- void onDetachedFromWindow ()
- void super_onDetachedFromWindow ()
- boolean onKeyLongPress (int keyCode, KeyEvent event)
- boolean super_onKeyLongPress (int keyCode, KeyEvent event)
- Dialog super_onCreateDialog (int id, Bundle args)
- void super_onPrepareDialog (int id, Dialog dialog, Bundle args)

### Protected Member Functions

- void onActivityResult (int requestCode, int resultCode, Intent data)
- void onApplyThemeResource (Theme theme, int resid, boolean first)
- void onChildTitleChanged (Activity childActivity, CharSequence title)
- Dialog onCreateDialog (int id)
- void onDestroy ()
- void onNewIntent (Intent intent)
- void onPause ()
- void onPostCreate (Bundle savedInstanceState)
- void onPostResume ()
- void onPrepareDialog (int id, Dialog dialog)
- void onRestart ()
- void onRestoreInstanceState (Bundle savedInstanceState)
- void onResume ()
- void onSaveInstanceState (Bundle outState)
- void onStart ()
- void onStop ()
- void onTitleChanged (CharSequence title, int color)
- void onUserLeaveHint ()
- Dialog onCreateDialog (int id, Bundle args)
- void onPrepareDialog (int id, Dialog dialog, Bundle args)

### Private Member Functions

- void loadApplication (Bundle loaderParams)
- void downloadUpgradeMinistro (String msg)
- void ministroNotFound ()
- void startApp (final boolean firstStart)

### Private Attributes

- ActivityInfo m_activityInfo = null
- DexClassLoader m_classLoader = null
- String[] m_qtLibs = null
- ServiceConnection m_ministroConnection

### Static Private Attributes

- static final int MINISTRO_INSTALL_REQUEST_CODE = 0xf3ee
- static final int MINISTRO_API_LEVEL =2
- static final int NECESSITAS_API_LEVEL =2
- static final String QT_PROVIDER ="necessitas"
- static final int QT_VERSION =0x040801
- static final String ERROR_CODE_KEY ="error.code"
- static final String ERROR_MESSAGE_KEY ="error.message"
- static final String DEX_PATH_KEY ="dex.path"
- static final String LIB_PATH_KEY ="lib.path"
- static final String LOADER_CLASS_NAME_KEY ="loader.class.name"
- static final String NATIVE_LIBRARIES_KEY ="native.libraries"
- static final String ENVIRONMENT_VARIABLES_KEY ="environment.variables"
- static final String APPLICATION_PARAMETERS_KEY ="application.parameters"
- static final String BUNDLED_LIBRARIES_KEY ="bundled.libraries"
- static final String MAIN_LIBRARY_KEY ="main.library"
- static final String NECESSITAS_API_LEVEL_KEY ="necessitas.api.level"
- static final String REQUIRED_MODULES_KEY ="required.modules"

  *Ministro server parameter keys.*
- static final String APPLICATION_TITLE_KEY ="application.title"
- static final String QT_PROVIDER_KEY ="qt.provider"
- static final String MINIMUM_MINISTRO_API_KEY ="minimum.ministro.api"
- static final String MINIMUM_QT_VERSION_KEY ="minimum.qt.version"
- static final String APPLICATION_PARAMETERS =null
- static final String ENVIRONMENT_VARIABLES ="QT_USE_ANDROID_NATIVE_STYLE=1\t"
- static final int INCOMPATIBLE_MINISTRO_VERSION =1

## 7.32.1 Detailed Description

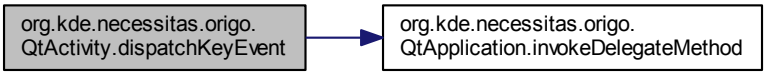
Definition at line 77 of file QtActivity.java.

## 7.32.2 Member Function Documentation

### 7.32.2.1 boolean org.kde.necessitas.origo.QtActivity.dispatchKeyEvent ( KeyEvent *event* ) [inline]

Definition at line 393 of file QtActivity.java.

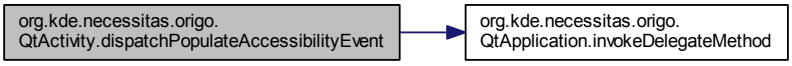
Here is the call graph for this function:



#### 7.32.2.2 boolean org.kde.necessitas.origo.QtActivity.dispatchPopulateAccessibilityEvent ( AccessibilityEvent *event* ) [inline]

Definition at line 407 of file QtActivity.java.

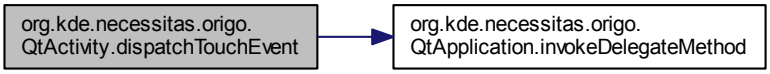
Here is the call graph for this function:




#### 7.32.2.3 boolean org.kde.necessitas.origo.QtActivity.dispatchTouchEvent ( MotionEvent *ev* ) [inline]

Definition at line 421 of file QtActivity.java.

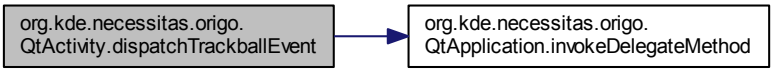
Here is the call graph for this function:




#### 7.32.2.4 boolean org.kde.necessitas.origo.QtActivity.dispatchTrackballEvent ( MotionEvent *ev* ) [inline]

Definition at line 435 of file QtActivity.java.

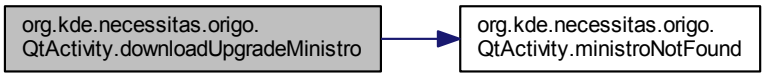
Here is the call graph for this function:



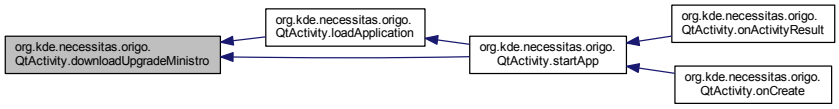

#### 7.32.2.5 void org.kde.necessitas.origo.QtActivity.downloadUpgradeMinistro ( String *msg* ) [inline],[private]

Definition at line 255 of file QtActivity.java.

Here is the call graph for this function:



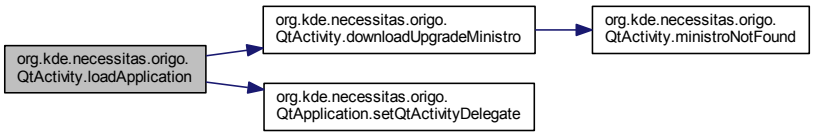

Here is the caller graph for this function:



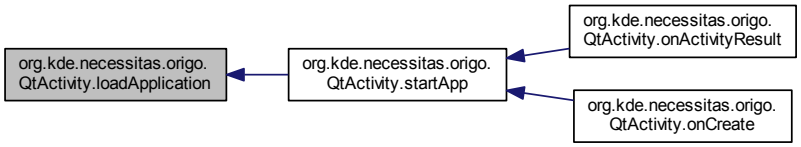

#### 7.32.2.6 void org.kde.necessitas.origo.QtActivity.loadApplication ( Bundle *loaderParams* ) [inline],[private]

Definition at line 127 of file QtActivity.java.

Here is the call graph for this function:



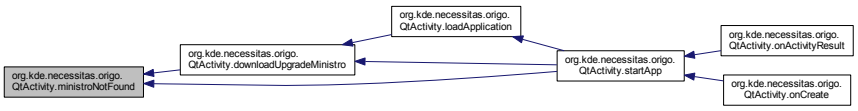

Here is the caller graph for this function:




#### 7.32.2.7 void org.kde.necessitas.origo.QtActivity.ministroNotFound ( ) [inline],[private]

Definition at line 284 of file QtActivity.java.

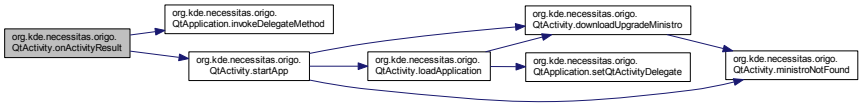
Here is the caller graph for this function:




#### 7.32.2.8 void org.kde.necessitas.origo.QtActivity.onActivityResult ( int *requestCode,* int *resultCode,* Intent *data* ) [inline], [protected]

Definition at line 449 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.9 void org.kde.necessitas.origo.QtActivity.onApplyThemeResource ( Theme *theme,* int *resid,* boolean *first* ) [inline], [protected]

Definition at line 468 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.10 void org.kde.necessitas.origo.QtActivity.onAttachedToWindow ( ) [inline]

Definition at line 1073 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.11 void org.kde.necessitas.origo.QtActivity.onBackPressed ( ) [inline]

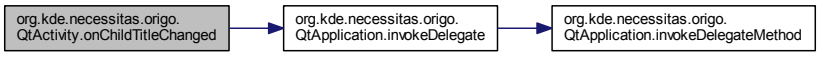
Definition at line 1085 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.12 void org.kde.necessitas.origo.QtActivity.onChildTitleChanged ( Activity *childActivity,* CharSequence *title* ) [inline],[protected]

Definition at line 481 of file QtActivity.java.

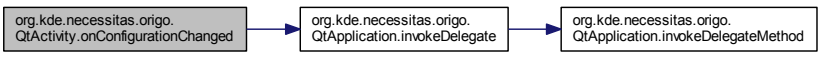
Here is the call graph for this function:




#### 7.32.2.13 void org.kde.necessitas.origo.QtActivity.onConfigurationChanged ( Configuration *newConfig* ) [inline]

Definition at line 493 of file QtActivity.java.

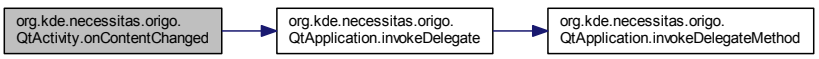
Here is the call graph for this function:




#### 7.32.2.14 void org.kde.necessitas.origo.QtActivity.onContentChanged ( ) [inline]

Definition at line 505 of file QtActivity.java.

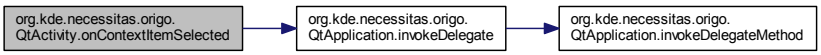
Here is the call graph for this function:



#### 7.32.2.15 boolean org.kde.necessitas.origo.QtActivity.onContextItemSelected ( MenuItem *item* ) [inline]

Definition at line 517 of file QtActivity.java.

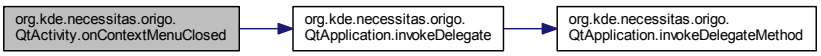
Here is the call graph for this function:




#### 7.32.2.16 void org.kde.necessitas.origo.QtActivity.onContextMenuClosed ( Menu *menu* ) [inline]

Definition at line 532 of file QtActivity.java.

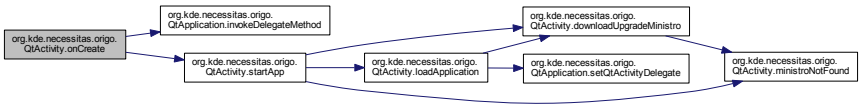
Here is the call graph for this function:




#### 7.32.2.17 void org.kde.necessitas.origo.QtActivity.onCreate ( Bundle *savedInstanceState* ) [inline]

Definition at line 544 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.18 void org.kde.necessitas.origo.QtActivity.onCreateContextMenu ( ContextMenu *menu,* View *v,* ContextMenuInfo *menuInfo* ) [inline]

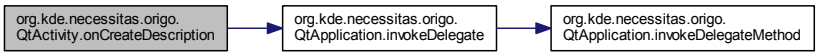
Definition at line 571 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.19 CharSequence org.kde.necessitas.origo.QtActivity.onCreateDescription ( ) [inline]

Definition at line 583 of file QtActivity.java.

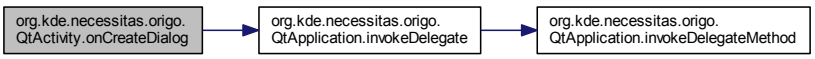
Here is the call graph for this function:




#### 7.32.2.20 Dialog org.kde.necessitas.origo.QtActivity.onCreateDialog ( int *id* ) [inline],[protected]

Definition at line 598 of file QtActivity.java.

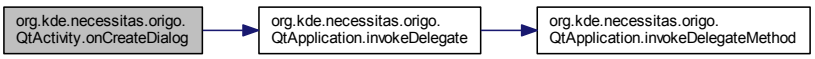
Here is the call graph for this function:




#### 7.32.2.21 Dialog org.kde.necessitas.origo.QtActivity.onCreateDialog ( int *id*, Bundle *args* ) [inline],[protected]

Definition at line 1126 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.22 boolean org.kde.necessitas.origo.QtActivity.onCreateOptionsMenu ( Menu *menu* ) [inline]

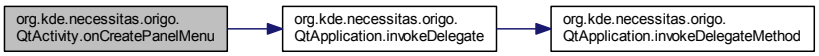
Definition at line 613 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.23 boolean org.kde.necessitas.origo.QtActivity.onCreatePanelMenu ( int *featureId*, Menu *menu* ) [inline]

Definition at line 628 of file QtActivity.java.

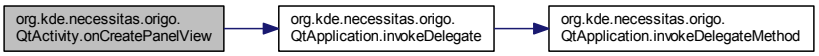
Here is the call graph for this function:




#### 7.32.2.24 View org.kde.necessitas.origo.QtActivity.onCreatePanelView ( int *featureId* ) [inline]

Definition at line 644 of file QtActivity.java.

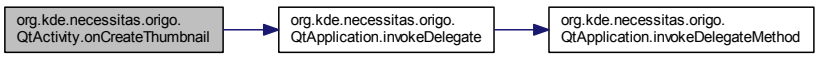
Here is the call graph for this function:




#### 7.32.2.25 boolean org.kde.necessitas.origo.QtActivity.onCreateThumbnail ( Bitmap *outBitmap*, Canvas *canvas* ) [inline]

Definition at line 659 of file QtActivity.java.

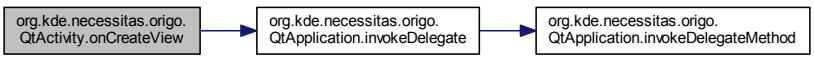
Here is the call graph for this function:




#### 7.32.2.26 View org.kde.necessitas.origo.QtActivity.onCreateView ( String *name*, Context *context*, AttributeSet *attrs* ) [inline]

Definition at line 674 of file QtActivity.java.

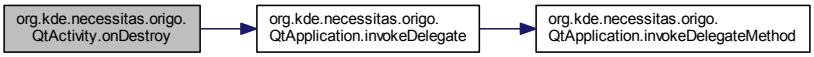
Here is the call graph for this function:



#### 7.32.2.27 void org.kde.necessitas.origo.QtActivity.onDestroy ( ) [inline],[protected]

Definition at line 689 of file QtActivity.java.

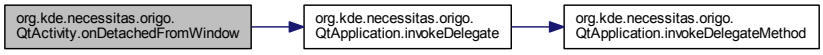
Here is the call graph for this function:




#### 7.32.2.28 void org.kde.necessitas.origo.QtActivity.onDetachedFromWindow ( ) [inline]

Definition at line 1097 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.29 boolean org.kde.necessitas.origo.QtActivity.onKeyDown ( int *keyCode*, KeyEvent *event* ) [inline]

Definition at line 698 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.30 boolean org.kde.necessitas.origo.QtActivity.onKeyLongPress ( int *keyCode*, KeyEvent *event* ) [inline]

Definition at line 1109 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.31 boolean org.kde.necessitas.origo.QtActivity.onKeyMultiple ( int *keyCode*, int *repeatCount*, KeyEvent *event* ) [inline]

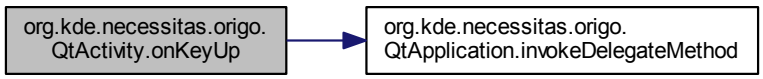
Definition at line 713 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.32 boolean org.kde.necessitas.origo.QtActivity.onKeyUp ( int *keyCode*, KeyEvent *event* ) [inline]

Definition at line 727 of file QtActivity.java.

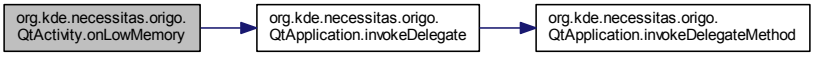
Here is the call graph for this function:




#### 7.32.2.33 void org.kde.necessitas.origo.QtActivity.onLowMemory ( ) [inline]

Definition at line 741 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.34 boolean org.kde.necessitas.origo.QtActivity.onMenuItemSelected ( int *featureId,* MenuItem *item* ) [inline]

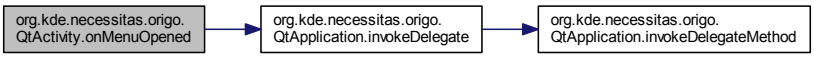
Definition at line 749 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.35 boolean org.kde.necessitas.origo.QtActivity.onMenuOpened ( int *featureId,* Menu *menu* ) [inline]

Definition at line 764 of file QtActivity.java.

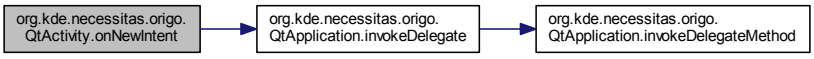
Here is the call graph for this function:




#### 7.32.2.36 void org.kde.necessitas.origo.QtActivity.onNewIntent ( Intent *intent* ) [inline], [protected]

Definition at line 779 of file QtActivity.java.

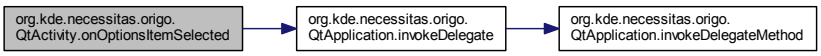
Here is the call graph for this function:




#### 7.32.2.37 boolean org.kde.necessitas.origo.QtActivity.onOptionsItemSelected ( MenuItem *item* ) [inline]

Definition at line 791 of file QtActivity.java.

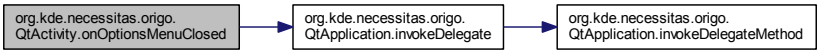
Here is the call graph for this function:




#### 7.32.2.38 void org.kde.necessitas.origo.QtActivity.onOptionsMenuClosed ( Menu *menu* ) [inline]

Definition at line 806 of file QtActivity.java.

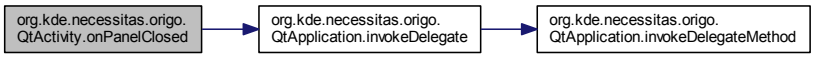
Here is the call graph for this function:




#### 7.32.2.39 void org.kde.necessitas.origo.QtActivity.onPanelClosed ( int *featureId,* Menu *menu* ) [inline]

Definition at line 818 of file QtActivity.java.

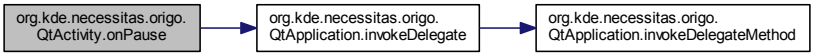
Here is the call graph for this function:




#### 7.32.2.40 void org.kde.necessitas.origo.QtActivity.onPause ( ) [inline], [protected]

Definition at line 830 of file QtActivity.java.

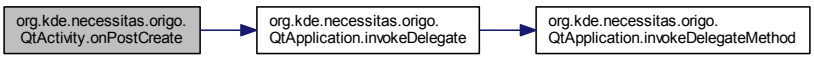
Here is the call graph for this function:




#### 7.32.2.41 void org.kde.necessitas.origo.QtActivity.onPostCreate ( Bundle *savedInstanceState* ) [inline], [protected]

Definition at line 838 of file QtActivity.java.

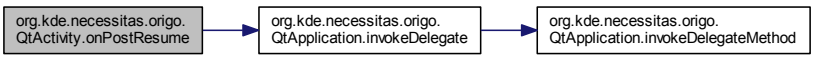
Here is the call graph for this function:




#### 7.32.2.42 void org.kde.necessitas.origo.QtActivity.onPostResume ( ) [inline], [protected]

Definition at line 846 of file QtActivity.java.

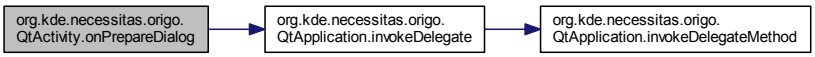
Here is the call graph for this function:




#### 7.32.2.43 void org.kde.necessitas.origo.QtActivity.onPrepareDialog ( int *id*, Dialog *dialog* ) [inline], [protected]

Definition at line 854 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.44 void org.kde.necessitas.origo.QtActivity.onPrepareDialog ( int *id*, Dialog *dialog*, Bundle *args* ) [inline], [protected]

Definition at line 1141 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.45 boolean org.kde.necessitas.origo.QtActivity.onPrepareOptionsMenu ( Menu *menu* ) [inline]

Definition at line 866 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.46 boolean org.kde.necessitas.origo.QtActivity.onPreparePanel ( int *featureId*, View *view*, Menu *menu* ) [inline]

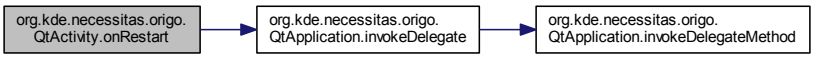
Definition at line 881 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.47 void org.kde.necessitas.origo.QtActivity.onRestart ( ) [inline],[protected]

Definition at line 896 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.48 void org.kde.necessitas.origo.QtActivity.onRestoreInstanceState ( Bundle *savedInstanceState* ) [inline], [protected]

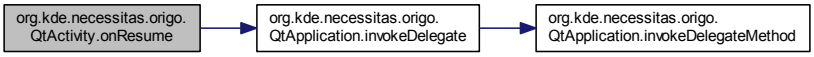
Definition at line 904 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.49 void org.kde.necessitas.origo.QtActivity.onResume ( ) [inline],[protected]

Definition at line 916 of file QtActivity.java.

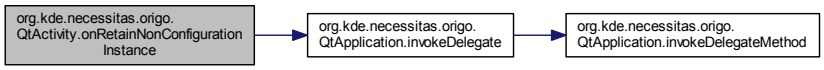
Here is the call graph for this function:




#### 7.32.2.50 Object org.kde.necessitas.origo.QtActivity.onRetainNonConfigurationInstance ( ) [inline]

Definition at line 924 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.51 void org.kde.necessitas.origo.QtActivity.onSaveInstanceState ( Bundle *outState* ) [inline], [protected]

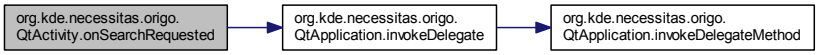
Definition at line 939 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.52 boolean org.kde.necessitas.origo.QtActivity.onSearchRequested ( ) [inline]

Definition at line 952 of file QtActivity.java.

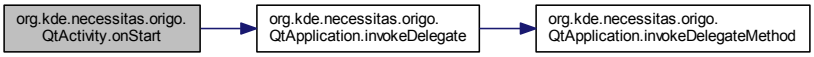
Here is the call graph for this function:




#### 7.32.2.53 void org.kde.necessitas.origo.QtActivity.onStart ( ) [inline], [protected]

Definition at line 967 of file QtActivity.java.

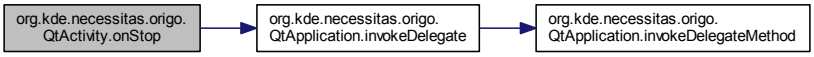
Here is the call graph for this function:




#### 7.32.2.54 void org.kde.necessitas.origo.QtActivity.onStop ( ) [inline], [protected]

Definition at line 975 of file QtActivity.java.

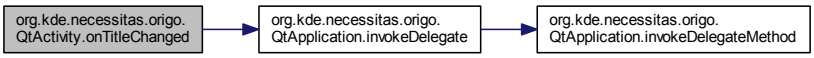
Here is the call graph for this function:




#### 7.32.2.55 void org.kde.necessitas.origo.QtActivity.onTitleChanged ( CharSequence *title*, int *color* ) [inline], [protected]

Definition at line 983 of file QtActivity.java.

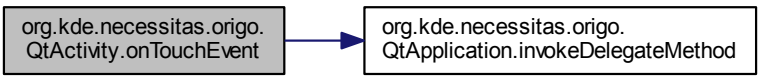
Here is the call graph for this function:




#### 7.32.2.56 boolean org.kde.necessitas.origo.QtActivity.onTouchEvent ( MotionEvent *event* ) [inline]

Definition at line 995 of file QtActivity.java.

Here is the call graph for this function:



#### 7.32.2.57 boolean org.kde.necessitas.origo.QtActivity.onTrackballEvent ( MotionEvent *event* ) [inline]

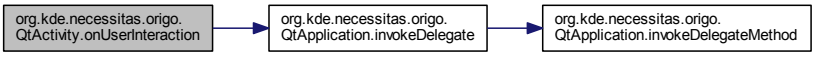
Definition at line 1009 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.58 void org.kde.necessitas.origo.QtActivity.onUserInteraction ( ) [inline]

Definition at line 1023 of file QtActivity.java.

Here is the call graph for this function:




#### 7.32.2.59 void org.kde.necessitas.origo.QtActivity.onUserLeaveHint ( ) [inline], [protected]

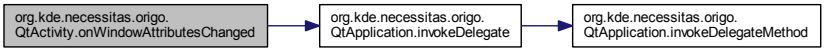
Definition at line 1035 of file QtActivity.java.

Here is the call graph for this function:

#### 7.32.2.60 void org.kde.necessitas.origo.QtActivity.onWindowAttributesChanged ( LayoutParams *params* ) [inline]

Definition at line 1047 of file QtActivity.java.

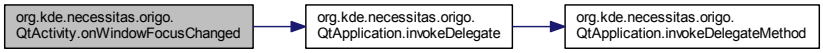
Here is the call graph for this function:



#### 7.32.2.61 void org.kde.necessitas.origo.QtActivity.onWindowFocusChanged ( boolean *hasFocus* ) [inline]

Definition at line 1059 of file QtActivity.java.

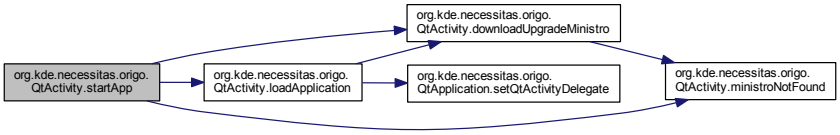
Here is the call graph for this function:



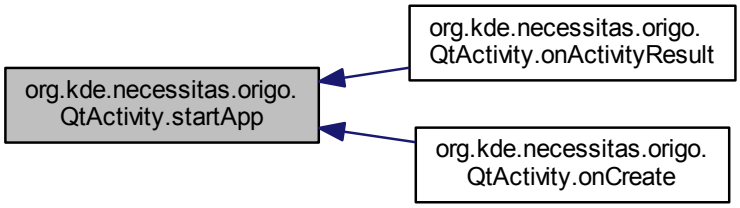

#### 7.32.2.62 void org.kde.necessitas.origo.QtActivity.startApp ( final boolean *firstStart* ) [inline], [private]

Definition at line 302 of file QtActivity.java.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.32.2.63 boolean org.kde.necessitas.origo.QtActivity.super_dispatchKeyEvent ( KeyEvent *event* ) [inline]

Definition at line 400 of file QtActivity.java.

#### 7.32.2.64 boolean org.kde.necessitas.origo.QtActivity.super_dispatchPopulateAccessibilityEvent ( AccessibilityEvent *event* ) [inline]

Definition at line 414 of file QtActivity.java.

#### 7.32.2.65 boolean org.kde.necessitas.origo.QtActivity.super_dispatchTouchEvent ( MotionEvent *event* ) [inline]

Definition at line 428 of file QtActivity.java.

#### 7.32.2.66 boolean org.kde.necessitas.origo.QtActivity.super_dispatchTrackballEvent ( MotionEvent *event* ) [inline]

Definition at line 442 of file QtActivity.java.

#### 7.32.2.67 void org.kde.necessitas.origo.QtActivity.super_onActivityResult ( int *requestCode,* int *resultCode,* Intent *data* ) [inline]

Definition at line 461 of file QtActivity.java.

#### 7.32.2.68 void org.kde.necessitas.origo.QtActivity.super_onApplyThemeResource ( Theme *theme,* int *resid,* boolean *first* ) [inline]

Definition at line 473 of file QtActivity.java.

#### 7.32.2.69 void org.kde.necessitas.origo.QtActivity.super_onAttachedToWindow ( ) [inline]

Definition at line 1078 of file QtActivity.java.

#### 7.32.2.70 void org.kde.necessitas.origo.QtActivity.super_onBackPressed ( ) [inline]

Definition at line 1090 of file QtActivity.java.

#### 7.32.2.71 void org.kde.necessitas.origo.QtActivity.super_onChildTitleChanged ( Activity *childActivity,* CharSequence *title* ) [inline]

Definition at line 486 of file QtActivity.java.

#### 7.32.2.72 void org.kde.necessitas.origo.QtActivity.super_onConfigurationChanged ( Configuration *newConfig* ) [inline]

Definition at line 498 of file QtActivity.java.

#### 7.32.2.73 void org.kde.necessitas.origo.QtActivity.super_onContentChanged ( ) [inline]

Definition at line 510 of file QtActivity.java.

#### 7.32.2.74 boolean org.kde.necessitas.origo.QtActivity.super_onContextItemSelected ( MenuItem *item* ) [inline]

Definition at line 525 of file QtActivity.java.

#### 7.32.2.75 void org.kde.necessitas.origo.QtActivity.super_onContextMenuClosed ( Menu *menu* ) [inline]

Definition at line 537 of file QtActivity.java.

#### 7.32.2.76 void org.kde.necessitas.origo.QtActivity.super_onCreateContextMenu ( ContextMenu *menu,* View *v,* ContextMenuInfo *menuInfo* ) [inline]

Definition at line 576 of file QtActivity.java.

#### 7.32.2.77 CharSequence org.kde.necessitas.origo.QtActivity.super_onCreateDescription ( ) [inline]

Definition at line 591 of file QtActivity.java.

#### 7.32.2.78 Dialog org.kde.necessitas.origo.QtActivity.super_onCreateDialog ( int *id* ) [inline]

Definition at line 606 of file QtActivity.java.

#### 7.32.2.79 Dialog org.kde.necessitas.origo.QtActivity.super_onCreateDialog ( int *id,* Bundle *args* ) [inline]

Definition at line 1134 of file QtActivity.java.

#### 7.32.2.80 boolean org.kde.necessitas.origo.QtActivity.super_onCreateOptionsMenu ( Menu *menu* ) [inline]

Definition at line 621 of file QtActivity.java.

#### 7.32.2.81 boolean org.kde.necessitas.origo.QtActivity.super_onCreatePanelMenu ( int *featureId,* Menu *menu* ) [inline]

Definition at line 636 of file QtActivity.java.

#### 7.32.2.82 View org.kde.necessitas.origo.QtActivity.super_onCreatePanelView ( int *featureId* ) [inline]

Definition at line 652 of file QtActivity.java.

#### 7.32.2.83 boolean org.kde.necessitas.origo.QtActivity.super_onCreateThumbnail ( Bitmap *outBitmap,* Canvas *canvas* ) [inline]

Definition at line 667 of file QtActivity.java.

#### 7.32.2.84 View org.kde.necessitas.origo.QtActivity.super_onCreateView ( String *name,* Context *context,* AttributeSet *attrs* ) [inline]

Definition at line 682 of file QtActivity.java.

#### 7.32.2.85 void org.kde.necessitas.origo.QtActivity.super_onDetachedFromWindow ( ) [inline]

Definition at line 1102 of file QtActivity.java.

#### 7.32.2.86 boolean org.kde.necessitas.origo.QtActivity.super_onKeyDown ( int *keyCode,* KeyEvent *event* ) [inline]

Definition at line 705 of file QtActivity.java.

#### 7.32.2.87 boolean org.kde.necessitas.origo.QtActivity.super_onKeyLongPress ( int *keyCode,* KeyEvent *event* ) [inline]

Definition at line 1116 of file QtActivity.java.

#### 7.32.2.88 boolean org.kde.necessitas.origo.QtActivity.super_onKeyMultiple ( int *keyCode,* int *repeatCount,* KeyEvent *event* ) [inline]

Definition at line 720 of file QtActivity.java.

#### 7.32.2.89 boolean org.kde.necessitas.origo.QtActivity.super_onKeyUp ( int *keyCode,* KeyEvent *event* ) [inline]

Definition at line 734 of file QtActivity.java.

#### 7.32.2.90 boolean org.kde.necessitas.origo.QtActivity.super_onMenuItemSelected ( int *featureId,* MenuItem *item* ) [inline]

Definition at line 757 of file QtActivity.java.

#### 7.32.2.91 boolean org.kde.necessitas.origo.QtActivity.super_onMenuOpened ( int *featureId,* Menu *menu* ) [inline]

Definition at line 772 of file QtActivity.java.

#### 7.32.2.92 void org.kde.necessitas.origo.QtActivity.super_onNewIntent ( Intent *intent* ) [inline]

Definition at line 784 of file QtActivity.java.

#### 7.32.2.93 boolean org.kde.necessitas.origo.QtActivity.super_onOptionsItemSelected ( MenuItem *item* ) [inline]

Definition at line 799 of file QtActivity.java.

#### 7.32.2.94 void org.kde.necessitas.origo.QtActivity.super_onOptionsMenuClosed ( Menu *menu* ) [inline]

Definition at line 811 of file QtActivity.java.

#### 7.32.2.95 void org.kde.necessitas.origo.QtActivity.super_onPanelClosed ( int *featureId,* Menu *menu* ) [inline]

Definition at line 823 of file QtActivity.java.

#### 7.32.2.96 void org.kde.necessitas.origo.QtActivity.super_onPrepareDialog ( int *id,* Dialog *dialog* ) [inline]

Definition at line 859 of file QtActivity.java.

#### 7.32.2.97 void org.kde.necessitas.origo.QtActivity.super_onPrepareDialog ( int *id,* Dialog *dialog,* Bundle *args* ) [inline]

Definition at line 1146 of file QtActivity.java.

#### 7.32.2.98 boolean org.kde.necessitas.origo.QtActivity.super_onPrepareOptionsMenu ( Menu *menu* ) [inline]

Definition at line 874 of file QtActivity.java.

#### 7.32.2.99 boolean org.kde.necessitas.origo.QtActivity.super_onPreparePanel ( int *featureId,* View *view,* Menu *menu* ) [inline]

Definition at line 889 of file QtActivity.java.

#### 7.32.2.100 void org.kde.necessitas.origo.QtActivity.super_onRestoreInstanceState ( Bundle *savedInstanceState* ) [inline]

Definition at line 909 of file QtActivity.java.

#### 7.32.2.101 Object org.kde.necessitas.origo.QtActivity.super_onRetainNonConfigurationInstance ( ) [inline]

Definition at line 932 of file QtActivity.java.

#### 7.32.2.102 void org.kde.necessitas.origo.QtActivity.super_onSaveInstanceState ( Bundle *outState* ) [inline]

Definition at line 944 of file QtActivity.java.

#### 7.32.2.103 boolean org.kde.necessitas.origo.QtActivity.super_onSearchRequested ( ) [inline]

Definition at line 960 of file QtActivity.java.

#### 7.32.2.104 void org.kde.necessitas.origo.QtActivity.super_onTitleChanged ( CharSequence *title,* int *color* ) [inline]

Definition at line 988 of file QtActivity.java.

#### 7.32.2.105 boolean org.kde.necessitas.origo.QtActivity.super_onTouchEvent ( MotionEvent *event* ) [inline]

Definition at line 1002 of file QtActivity.java.

#### 7.32.2.106 boolean org.kde.necessitas.origo.QtActivity.super_onTrackballEvent ( MotionEvent *event* ) [inline]

Definition at line 1016 of file QtActivity.java.

#### 7.32.2.107 void org.kde.necessitas.origo.QtActivity.super_onUserInteraction ( ) [inline]

Definition at line 1028 of file QtActivity.java.

#### 7.32.2.108 void org.kde.necessitas.origo.QtActivity.super_onUserLeaveHint ( ) [inline]

Definition at line 1040 of file QtActivity.java.

#### 7.32.2.109 void org.kde.necessitas.origo.QtActivity.super_onWindowAttributesChanged ( LayoutParams *params* ) [inline]

Definition at line 1052 of file QtActivity.java.

#### 7.32.2.110 void org.kde.necessitas.origo.QtActivity.super_onWindowFocusChanged ( boolean *hasFocus* ) [inline]

Definition at line 1064 of file QtActivity.java.

### 7.32.3 Member Data Documentation

#### 7.32.3.1 final String org.kde.necessitas.origo.QtActivity.APPLICATION_PARAMETERS =null [static], [private]

Definition at line 109 of file QtActivity.java.

#### 7.32.3.2 final String org.kde.necessitas.origo.QtActivity.APPLICATION_PARAMETERS_KEY ="application.parameters" [static], [private]

Definition at line 92 of file QtActivity.java.

#### 7.32.3.3 final String org.kde.necessitas.origo.QtActivity.APPLICATION_TITLE_KEY ="application.title" [static], [private]

Definition at line 99 of file QtActivity.java.

#### 7.32.3.4 final String org.kde.necessitas.origo.QtActivity.BUNDLED_LIBRARIES_KEY ="bundled.libraries" [static], [private]

Definition at line 93 of file QtActivity.java.

#### 7.32.3.5 final String org.kde.necessitas.origo.QtActivity.DEX_PATH_KEY ="dex.path" [static], [private]

Definition at line 87 of file QtActivity.java.

#### 7.32.3.6 final String org.kde.necessitas.origo.QtActivity.ENVIRONMENT_VARIABLES ="QT_USE_ANDROID_NATIVE_STYLE=1\t" [static], [private]

Definition at line 114 of file QtActivity.java.

#### 7.32.3.7 final String org.kde.necessitas.origo.QtActivity.ENVIRONMENT_VARIABLES_KEY ="environment.variables" [static], [private]

Definition at line 91 of file QtActivity.java.

#### 7.32.3.8 final String org.kde.necessitas.origo.QtActivity.ERROR_CODE_KEY ="error.code" [static], [private]

Definition at line 85 of file QtActivity.java.

#### 7.32.3.9 final String org.kde.necessitas.origo.QtActivity.ERROR_MESSAGE_KEY ="error.message" [static], [private]

Definition at line 86 of file QtActivity.java.

#### 7.32.3.10 final int org.kde.necessitas.origo.QtActivity.INCOMPATIBLE_MINISTRO_VERSION =1 [static], [private]

Definition at line 121 of file QtActivity.java.

#### 7.32.3.11 final String org.kde.necessitas.origo.QtActivity.LIB_PATH_KEY ="lib.path" [static], [private]

Definition at line 88 of file QtActivity.java.

#### 7.32.3.12 final String org.kde.necessitas.origo.QtActivity.LOADER_CLASS_NAME_KEY ="loader.class.name" [static], [private]

Definition at line 89 of file QtActivity.java.

#### 7.32.3.13 ActivityInfo org.kde.necessitas.origo.QtActivity.m_activityInfo = null [private]

Definition at line 122 of file QtActivity.java.

#### 7.32.3.14 DexClassLoader org.kde.necessitas.origo.QtActivity.m_classLoader = null [private]

Definition at line 123 of file QtActivity.java.

#### 7.32.3.15 ServiceConnection org.kde.necessitas.origo.QtActivity.m_ministroConnection [private]

Definition at line 207 of file QtActivity.java.

#### 7.32.3.16 String [] org.kde.necessitas.origo.QtActivity.m_qtLibs = null [private]

Definition at line 124 of file QtActivity.java.

#### 7.32.3.17 final String org.kde.necessitas.origo.QtActivity.MAIN_LIBRARY_KEY ="main.library" [static], [private]

Definition at line 94 of file QtActivity.java.

#### 7.32.3.18 final String org.kde.necessitas.origo.QtActivity.MINIMUM_MINISTRO_API_KEY ="minimum.ministro.api" [static], [private]

Definition at line 101 of file QtActivity.java.

#### 7.32.3.19 final String org.kde.necessitas.origo.QtActivity.MINIMUM_QT_VERSION_KEY ="minimum.qt.version" [static], [private]

Definition at line 102 of file QtActivity.java.

#### 7.32.3.20 final int org.kde.necessitas.origo.QtActivity.MINISTRO_API_LEVEL =2 [static], [private]

Definition at line 80 of file QtActivity.java.

#### 7.32.3.21 final int org.kde.necessitas.origo.QtActivity.MINISTRO_INSTALL_REQUEST_CODE = 0xf3ee [static], [private]

Definition at line 79 of file QtActivity.java.

#### 7.32.3.22 final String org.kde.necessitas.origo.QtActivity.NATIVE_LIBRARIES_KEY ="native.libraries" [static], [private]

Definition at line 90 of file QtActivity.java.

#### 7.32.3.23 final int org.kde.necessitas.origo.QtActivity.NECESSITAS_API_LEVEL =2 [static], [private]

Definition at line 81 of file QtActivity.java.

#### 7.32.3.24 final String org.kde.necessitas.origo.QtActivity.NECESSITAS_API_LEVEL_KEY ="necessitas.api.level" [static], [private]

Definition at line 95 of file QtActivity.java.

#### 7.32.3.25 final String org.kde.necessitas.origo.QtActivity.QT_PROVIDER ="necessitas" [static], [private]

Definition at line 82 of file QtActivity.java.

#### 7.32.3.26 final String org.kde.necessitas.origo.QtActivity.QT_PROVIDER_KEY ="qt.provider" [static], [private]

Definition at line 100 of file QtActivity.java.

#### 7.32.3.27 final int org.kde.necessitas.origo.QtActivity.QT_VERSION =0x040801 [static], [private]

Definition at line 83 of file QtActivity.java.

#### 7.32.3.28 final String org.kde.necessitas.origo.QtActivity.REQUIRED_MODULES_KEY ="required.modules" [static], [private]

Ministro server parameter keys.

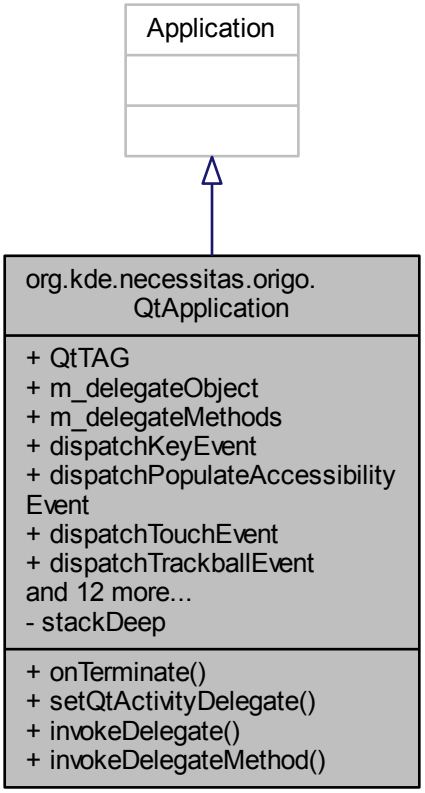
Definition at line 98 of file QtActivity.java.

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

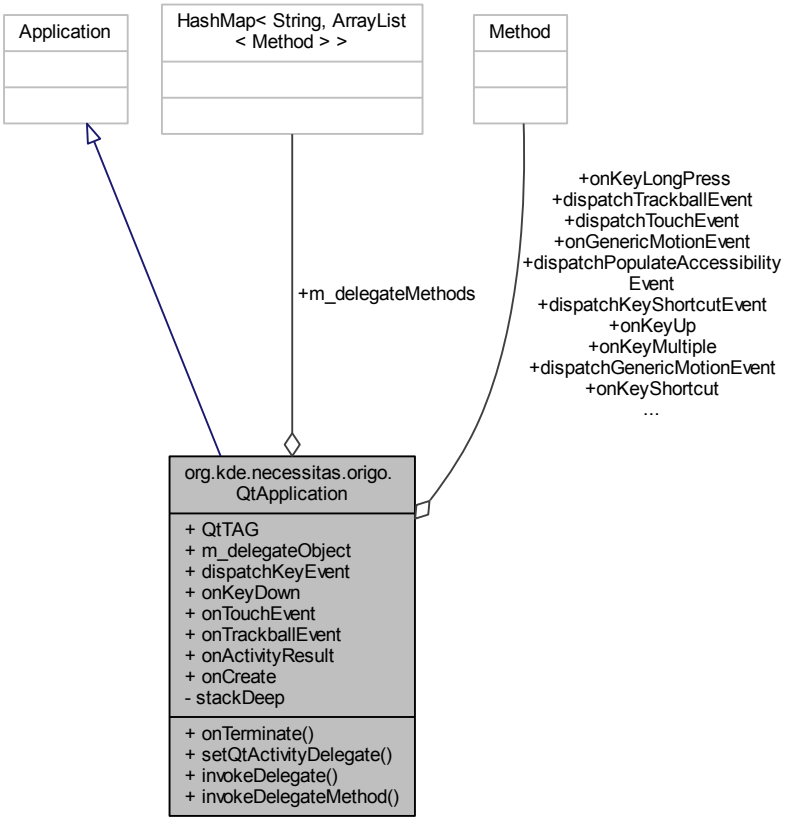
- C++/customIrcClient/android/src/org/kde/necessitas/origo/QtActivity.java

## 7.33 org.kde.necessitas.origo.QtApplication Class Reference

Inheritance diagram for org.kde.necessitas.origo.QtApplication:



Collaboration diagram for org.kde.necessitas.origo.QtApplication:




## Classes

- class **InvokeResult**

## Public Member Functions

- void onTerminate ()

## Static Public Member Functions

- static void setQtActivityDelegate (Object listener)
- static InvokeResult invokeDelegate (Object...args)
- static Object invokeDelegateMethod (Method m, Object...args)

## Static Public Attributes

- static final String QtTAG ="Qt"
- static Object m_delegateObject = null
- static HashMap< String, ArrayList< Method > > m_delegateMethods = new HashMap<String, ArrayList<Method>>()

- static Method dispatchKeyEvent = null
- static Method dispatchPopulateAccessibilityEvent = null
- static Method dispatchTouchEvent = null
- static Method dispatchTrackballEvent = null
- static Method onKeyDown = null
- static Method onKeyMultiple = null
- static Method onKeyUp = null
- static Method onTouchEvent = null
- static Method onTrackballEvent = null
- static Method onActivityResult = null
- static Method onCreate = null
- static Method onKeyLongPress = null
- static Method dispatchKeyShortcutEvent = null
- static Method onKeyShortcut = null
- static Method dispatchGenericMotionEvent = null
- static Method onGenericMotionEvent = null

### Static Private Attributes

- static int stackDeep =-1

## 7.33.1 Detailed Description

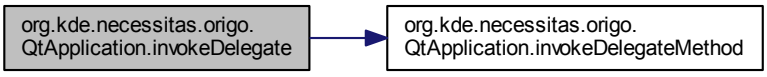
Definition at line 37 of file QtApplication.java.

## 7.33.2 Member Function Documentation

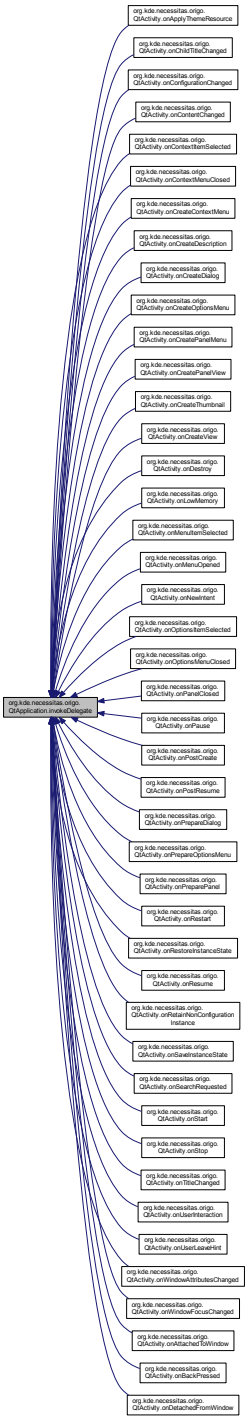
#### 7.33.2.1 static InvokeResult org.kde.necessitas.origo.QtApplication.invokeDelegate ( Object... *args* ) [inline], [static]

Definition at line 171 of file QtApplication.java.

Here is the call graph for this function:



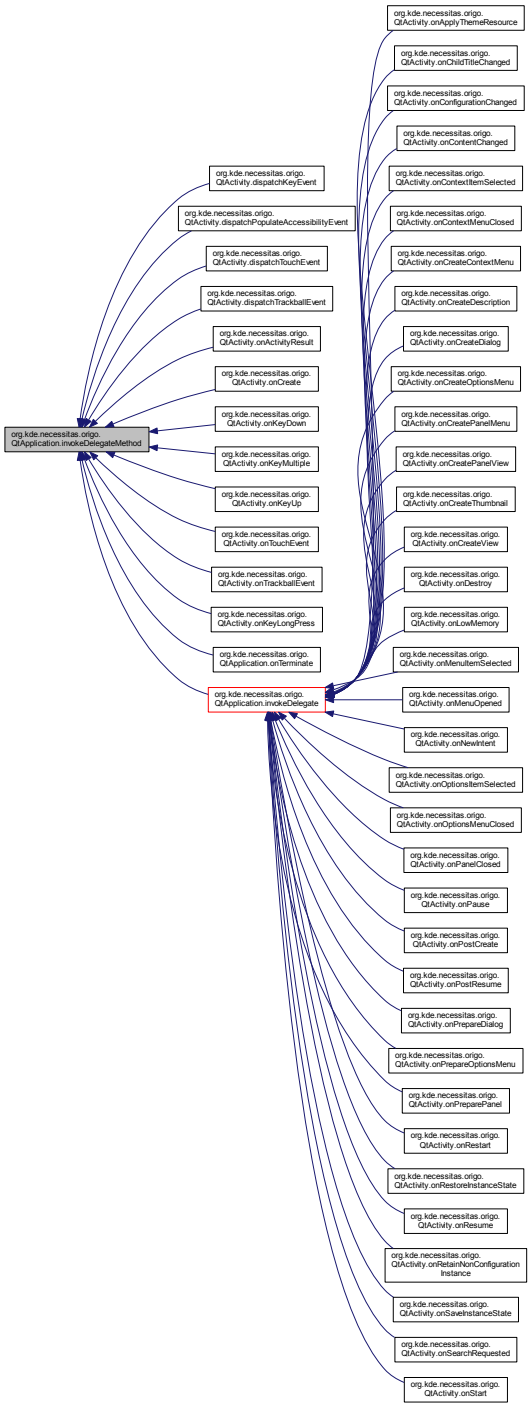
Here is the caller graph for this function:




#### 7.33.2.2 static Object org.kde.necessitas.origo.QtApplication.invokeDelegateMethod ( Method *m*, Object... *args* ) [inline],[static]

Definition at line 201 of file QtApplication.java.

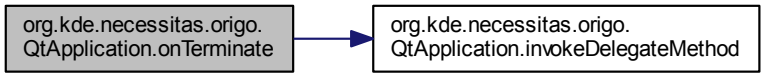
Here is the caller graph for this function:




#### 7.33.2.3 void org.kde.necessitas.origo.QtApplication.onTerminate ( ) [inline]

Definition at line 158 of file QtApplication.java.

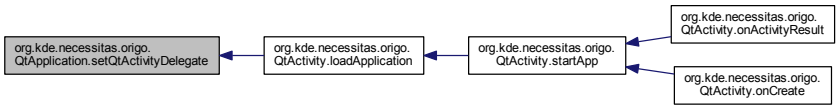
Here is the call graph for this function:




#### 7.33.2.4 static void org.kde.necessitas.origo.QtApplication.setQtActivityDelegate ( Object *listener* ) [inline], [static]

Definition at line 113 of file QtApplication.java.

Here is the caller graph for this function:




### 7.33.3 Member Data Documentation

#### 7.33.3.1 Method org.kde.necessitas.origo.QtApplication.dispatchGenericMotionEvent = null [static]

Definition at line 56 of file QtApplication.java.

#### 7.33.3.2 Method org.kde.necessitas.origo.QtApplication.dispatchKeyEvent = null [static]

Definition at line 42 of file QtApplication.java.

#### 7.33.3.3 Method org.kde.necessitas.origo.QtApplication.dispatchKeyShortcutEvent = null [static]

Definition at line 54 of file QtApplication.java.

#### 7.33.3.4 Method org.kde.necessitas.origo.QtApplication.dispatchPopulateAccessibilityEvent = null [static]

Definition at line 43 of file QtApplication.java.

#### 7.33.3.5 Method org.kde.necessitas.origo.QtApplication.dispatchTouchEvent = null [static]

Definition at line 44 of file QtApplication.java.

#### 7.33.3.6 Method org.kde.necessitas.origo.QtApplication.dispatchTrackballEvent = null [static]

Definition at line 45 of file QtApplication.java.

#### 7.33.3.7 HashMap<String, ArrayList<Method> > org.kde.necessitas.origo.QtApplication.m_delegateMethods = new HashMap<String, ArrayList<Method>>() [static]

Definition at line 41 of file QtApplication.java.

#### 7.33.3.8 Object org.kde.necessitas.origo.QtApplication.m_delegateObject = null [static]

Definition at line 40 of file QtApplication.java.

#### 7.33.3.9 Method org.kde.necessitas.origo.QtApplication.onActivityResult = null [static]

Definition at line 51 of file QtApplication.java.

#### 7.33.3.10 Method org.kde.necessitas.origo.QtApplication.onCreate = null [static]

Definition at line 52 of file QtApplication.java.

#### 7.33.3.11 Method org.kde.necessitas.origo.QtApplication.onGenericMotionEvent = null [static]

Definition at line 57 of file QtApplication.java.

#### 7.33.3.12 Method org.kde.necessitas.origo.QtApplication.onKeyDown = null [static]

Definition at line 46 of file QtApplication.java.

#### 7.33.3.13 Method org.kde.necessitas.origo.QtApplication.onKeyLongPress = null [static]

Definition at line 53 of file QtApplication.java.

#### 7.33.3.14 Method org.kde.necessitas.origo.QtApplication.onKeyMultiple = null [static]

Definition at line 47 of file QtApplication.java.

#### 7.33.3.15 Method org.kde.necessitas.origo.QtApplication.onKeyShortcut = null [static]

Definition at line 55 of file QtApplication.java.

#### 7.33.3.16 Method org.kde.necessitas.origo.QtApplication.onKeyUp = null [static]

Definition at line 48 of file QtApplication.java.

#### 7.33.3.17 Method org.kde.necessitas.origo.QtApplication.onTouchEvent = null [static]

Definition at line 49 of file QtApplication.java.

#### 7.33.3.18 Method org.kde.necessitas.origo.QtApplication.onTrackballEvent = null [static]

Definition at line 50 of file QtApplication.java.

#### 7.33.3.19 final String org.kde.necessitas.origo.QtApplication.QtTAG ="Qt" [static]

Definition at line 39 of file QtApplication.java.

#### 7.33.3.20 int org.kde.necessitas.origo.QtApplication.stackDeep =-1 [static],[private]

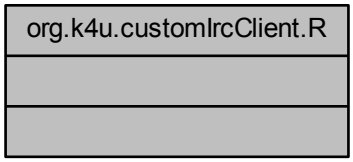
Definition at line 170 of file QtApplication.java.

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

- C++/customIrcClient/android/src/org/kde/necessitas/origo/QtApplication.java

## 7.34 org.k4u.customIrcClient.R Class Reference

Collaboration diagram for org.k4u.customIrcClient.R:




### Classes

- class **array**
- class **attr**
- class **drawable**
- class **layout**
- class **string**

### 7.34.1 Detailed Description

Definition at line 10 of file R.java.

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

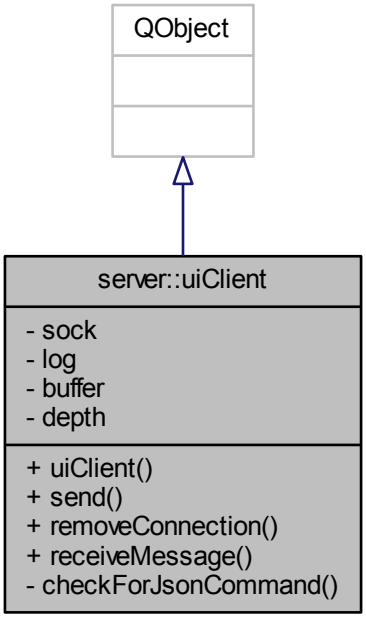
- C++/customIrcClient/android/gen/org/k4u/customIrcClient/R.java

## 7.35 server::uiClient Class Reference

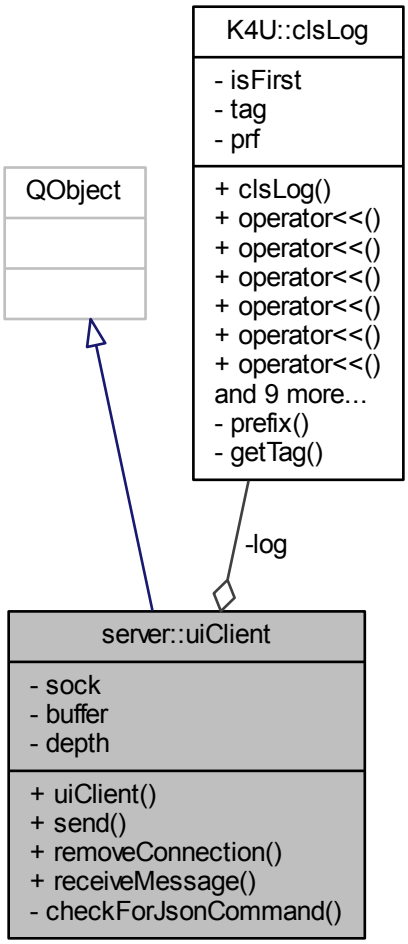
The uiClient class.

```
#include <uiclient.h>
```

Inheritance diagram for server::uiClient:



Collaboration diagram for server::uiClient:




## Public Slots

- void removeConnection ()
- void receiveMessage ()

## Signals

- void disconnected (uiClient *client)

  *Emitted when the user client disconnects.*
- void chatReceived (QString message)

  *Emitted when the client sends a chat.*
- void userQuery (uiClient *client)

  *Emitted when the client wants all the users.*
- void getEvents (uiClient *client)

  *Emitted when the client wants all the events.*

- void changeOwnUser (jsonCommand &comm)

  *Emitted when the clients wants to change our own user.*

### Public Member Functions

- uiClient (QTcpSocket *parent)

  *uiClient constructor*
- void send (QString data)

  *Sends data to the socket.*

### Private Member Functions

- bool checkForJsonCommand (QByteArray toAdd, jsonCommand &comm)

  *Adds data to buffer and checks for json commands.*

### Private Attributes

- QTcpSocket * sock
- clsLog log
- QString * buffer
- int depth

## 7.35.1 Detailed Description

The uiClient class.

**Author**

Koen Beckers (K-4U)

Definition at line 17 of file uiclient.h.

## 7.35.2 Constructor & Destructor Documentation

### 7.35.2.1 server::uiClient::uiClient ( QTcpSocket * *parent* )

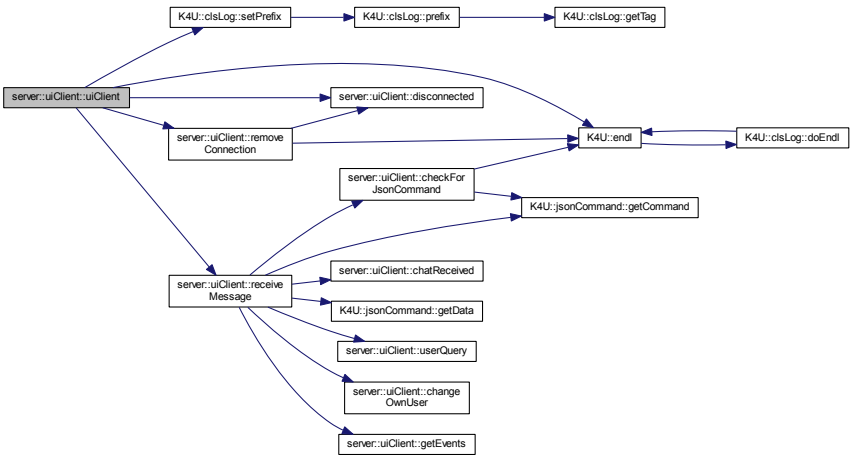
uiClient constructor

**Parameters**

| | |
|---|---|
| *parent* | The socket that the instance should handle |

Definition at line 5 of file uiclient.cpp.

Here is the call graph for this function:




### 7.35.3 Member Function Documentation

#### 7.35.3.1 void server::uiClient::changeOwnUser ( jsonCommand & *comm* ) [signal]

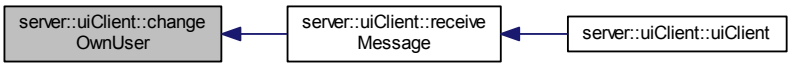
Emitted when the clients wants to change our own user.

**Parameters**

| | |
|---|---|
| *comm* | A jsonCommand with the data. |

**Todo** Replace the comm object with a QString or NickAndStatus!

Here is the caller graph for this function:



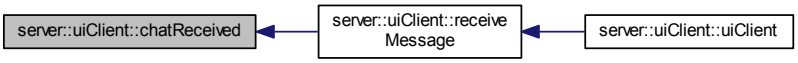

#### 7.35.3.2 void server::uiClient::chatReceived ( QString *message* ) [signal]

Emitted when the client sends a chat.

**Parameters**

| | |
|---|---|
| *message* | The message that the client wants to chat |

Here is the caller graph for this function:




#### 7.35.3.3 bool server::uiClient::checkForJsonCommand ( QByteArray *toAdd*, jsonCommand & *comm* ) [private]

Adds data to buffer and checks for json commands.

**Parameters**

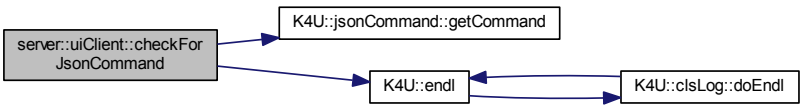
| | |
|---|---|
| *toAdd* | The bytes to add |
| *comm* | Pointer to a jsonCommand to be filled |

**Returns**

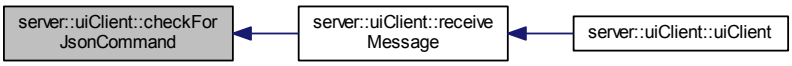
true if a jsonCommand was detected Constantly looks for a JSON command in the buffer This is done with the depth. Every time a { or [ is detected, depth is raised. Every time a } or ] is detected, depth is decreased. When depth is 0, there should be a complete json command!

Definition at line 32 of file uiclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:



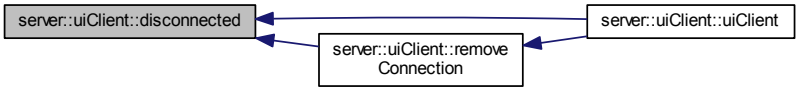

#### 7.35.3.4 void server::uiClient::disconnected ( uiClient * *client* ) [signal]

Emitted when the user client disconnects.

**Parameters**

| | |
|---|---|
| *client* | The instance of this class |

Here is the caller graph for this function:



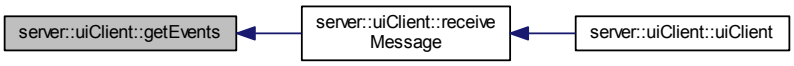

#### 7.35.3.5 void server::uiClient::getEvents ( uiClient * client ) [signal]

Emitted when the client wants all the events.

**Parameters**

| | |
|---|---|
| *client* | The instance of this class |

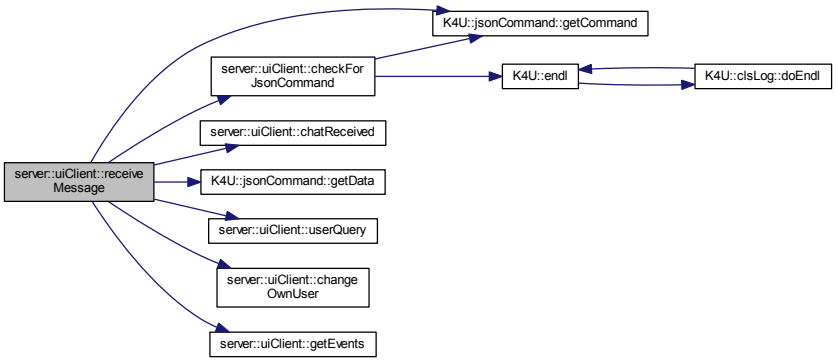
Here is the caller graph for this function:



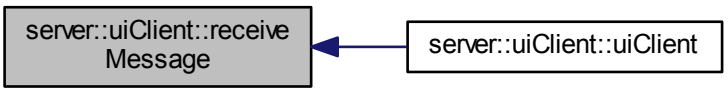

#### 7.35.3.6 void server::uiClient::receiveMessage ( ) [slot]

Definition at line 55 of file uiclient.cpp.

Here is the call graph for this function:



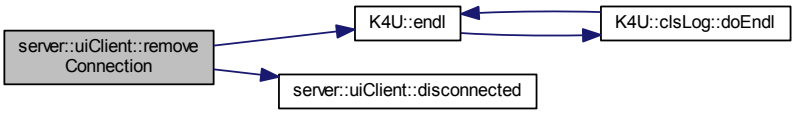
Here is the caller graph for this function:




#### 7.35.3.7 void server::uiClient::removeConnection ( ) [slot]

Definition at line 25 of file uiclient.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.35.3.8 void server::uiClient::send ( QString *data* )

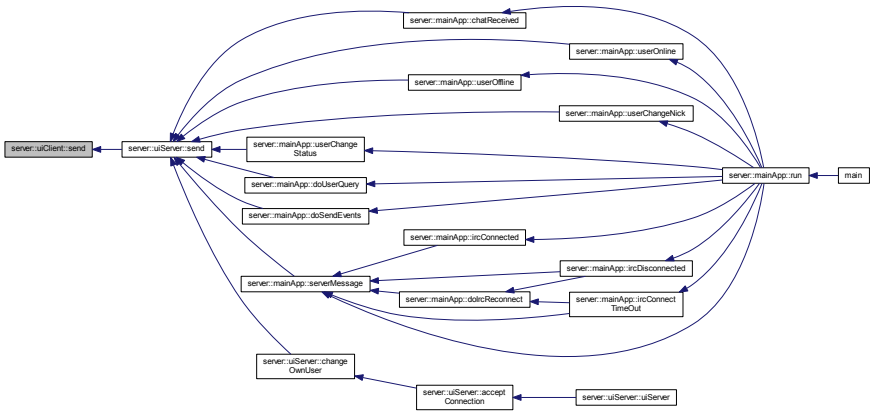
Sends data to the socket.

**Parameters**

| | |
|---|---|
| *data* | The data to send |

Definition at line 21 of file uiclient.cpp.

Here is the caller graph for this function:




#### 7.35.3.9 void server::uiClient::userQuery ( uiClient * *client* ) [signal]

Emitted when the client wants all the users.

**Parameters**

| | |
|---|---|
| *client* | The instance of this class |

Here is the caller graph for this function:

### 7.35.4 Member Data Documentation

#### 7.35.4.1 QString* server::uiClient::buffer [private]

Definition at line 83 of file uiclient.h.

#### 7.35.4.2 int server::uiClient::depth [private]

Definition at line 84 of file uiclient.h.

#### 7.35.4.3 clsLog server::uiClient::log [private]

Definition at line 82 of file uiclient.h.

#### 7.35.4.4 QTcpSocket* server::uiClient::sock [private]

Definition at line 81 of file uiclient.h.

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

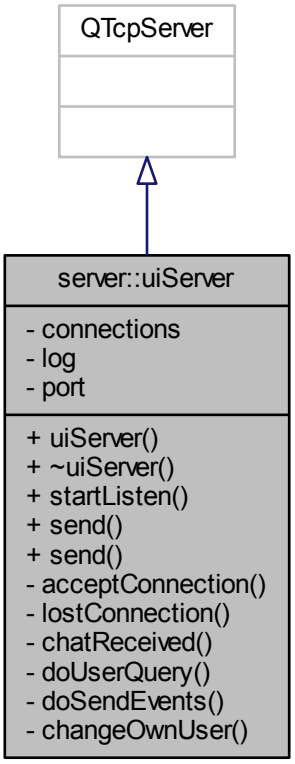
- C++/customIrcServer/uiclient.h
- C++/customIrcServer/uiclient.cpp

## 7.36 server::uiServer Class Reference

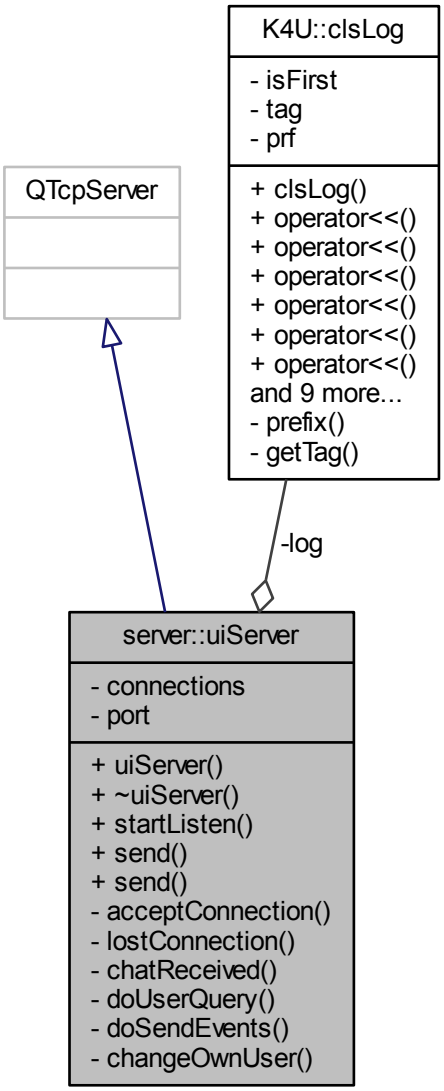
The uiServer class.

```
#include <uiserver.h>
```

Inheritance diagram for server::uiServer:



Collaboration diagram for server::uiServer:




## Signals

- void sgnChatReceived (QString message)

  *Emitted when a chatmessage has been send from a client.*
- void sgnUserQuery (uiClient *client)

  *Emitted when a client wants all the users.*
- void sgnSendEvents (uiClient *client)

  *Emitted when a client wants all the events.*
- void sgnChangeOwnUserStatus (QString newStatus)

  *Emitted when a client wants to change our own status.*

### Public Member Functions

- uiServer (int port=DEFAULT_LISTENING_PORT, QObject *parent=0)

  *uiServer constructor*
- ~uiServer ()
- void startListen ()

  *Starts to listen on the port.*
- void send (const jsonCommand &toSend)

  *Sends a jsonCommand to all the clients.*
- void send (uiClient *client, const jsonCommand &toSend)

  *Sends a jsonCommand to given client.*

### Private Slots

- void acceptConnection ()

  *Triggered when a client connects.*
- void lostConnection (uiClient *client)

  *Triggered when a client disconnects.*
- void chatReceived (QString message)

  *Triggered when a client sends a chatmessage.*
- void doUserQuery (uiClient *client)

  *Triggered when a client wants all the users.*
- void doSendEvents (uiClient *client)

  *Triggered when a client wants all the events.*
- void changeOwnUser (jsonCommand &comm)

  *Triggered when a client wants to change something about our own user.*

### Private Attributes

- QList< uiClient * > connections
- clsLog log
- int port

## 7.36.1 Detailed Description

The uiServer class.

**Author**

Koen Beckers (K-4U) Listens for incoming connections

Definition at line 21 of file uiserver.h.

## 7.36.2 Constructor & Destructor Documentation

### 7.36.2.1 server::uiServer::uiServer ( int *port* = DEFAULT_LISTENING_PORT, QObject * *parent* = 0 )

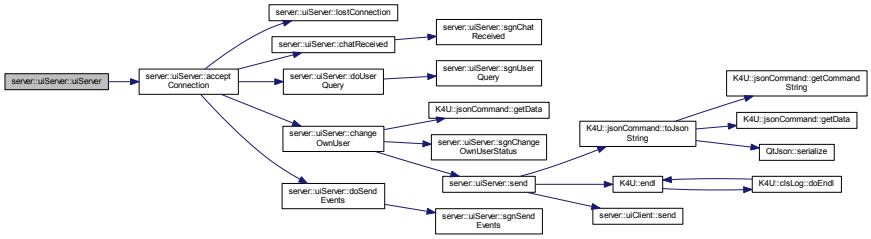
uiServer constructor

**Parameters**

| | |
|---|---|
| *port* | The port to listen on, defaults to DEFAULT_LISTENING_PORT |
| *parent* | The QT parent object |

Definition at line 8 of file uiserver.cpp.

Here is the call graph for this function:




#### 7.36.2.2 server::uiServer::~uiServer ( )

Definition at line 16 of file uiserver.cpp.

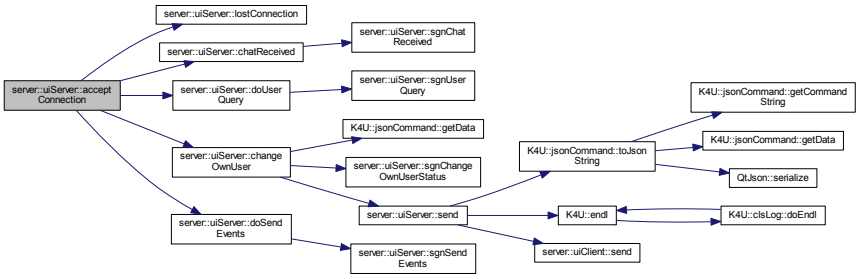
### 7.36.3 Member Function Documentation

#### 7.36.3.1 void server::uiServer::acceptConnection ( ) [private],[slot]

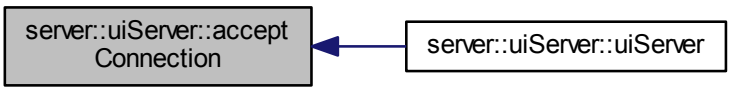
Triggered when a client connects.

Definition at line 40 of file uiserver.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:



#### 7.36.3.2 void server::uiServer::changeOwnUser ( jsonCommand & *comm* ) [private],[slot]

Triggered when a client wants to change something about our own user.

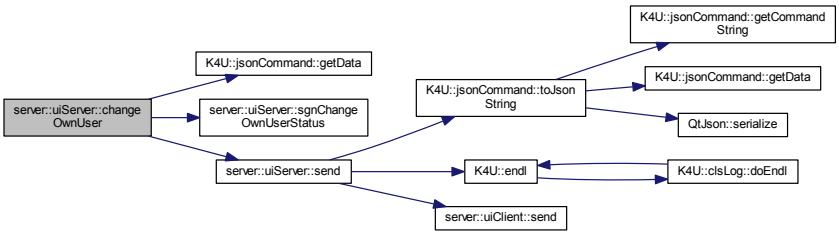
**Parameters**

| | |
|---|---|
| *comm* | The jsonCommand holding the data |

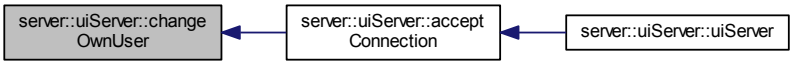
**Todo** Replace jsonCommand by something else

Definition at line 68 of file uiserver.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.36.3.3 void server::uiServer::chatReceived ( QString *message* ) [private],[slot]

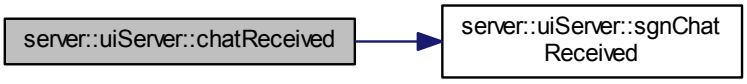
Triggered when a client sends a chatmessage.

**Parameters**

| | |
|---|---|
| *message* | The chatmessage received |

Definition at line 56 of file uiserver.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

#### 7.36.3.4 void server::uiServer::doSendEvents ( uiClient * client ) [private],[slot]

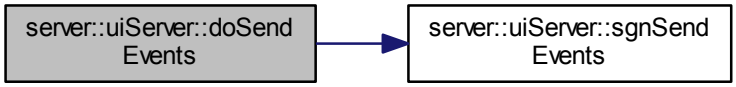
Triggered when a client wants all the events.

**Parameters**

| | |
|---|---|
| *client* | The client instance that wants the events |

Definition at line 64 of file uiserver.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.36.3.5 void server::uiServer::doUserQuery ( uiClient * client ) [private],[slot]

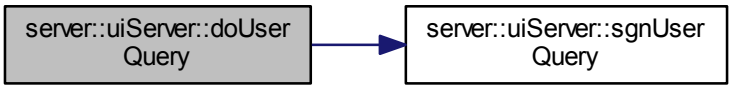
Triggered when a client wants all the users.

**Parameters**

| | |
|---|---|
| *client* | The client instance that wants the users |

Definition at line 60 of file uiserver.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:

#### 7.36.3.6 void server::uiServer::lostConnection ( uiClient * *client* ) [private],[slot]

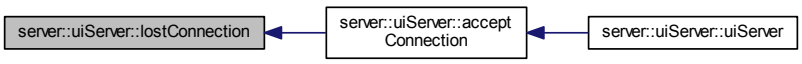
Triggered when a client disconnects.

**Parameters**

| | |
|---|---|
| *client* | The client instance that disconnected |

Definition at line 52 of file uiserver.cpp.

Here is the caller graph for this function:




#### 7.36.3.7 void server::uiServer::send ( const jsonCommand & *toSend* )

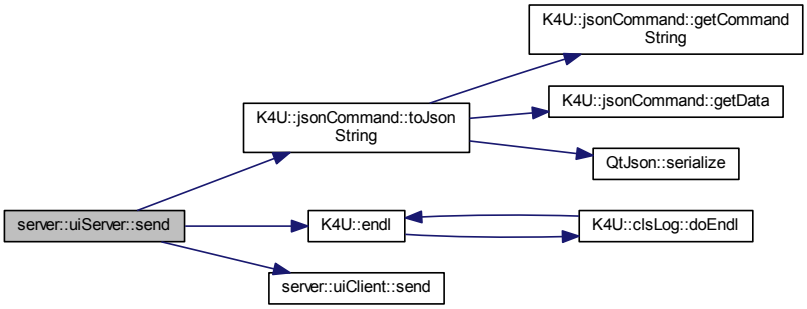
Sends a jsonCommand to all the clients.

**Parameters**

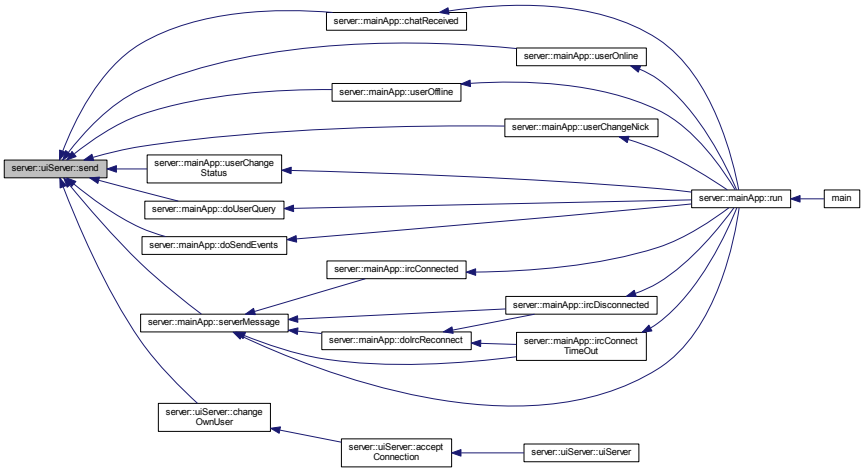
| | |
|---|---|
| *toSend* | The command to send |

Definition at line 24 of file uiserver.cpp.

Here is the call graph for this function:




Here is the caller graph for this function:




#### 7.36.3.8 void server::uiServer::send ( uiClient * *client*, const jsonCommand & *toSend* )

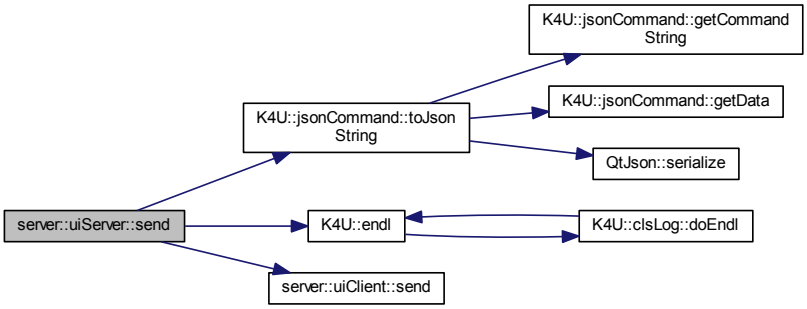
Sends a jsonCommand to given client.

**Parameters**

| | |
|---|---|
| *client* | The client to send the command to |
| *toSend* | The command to send |

Definition at line 33 of file uiserver.cpp.

Here is the call graph for this function:



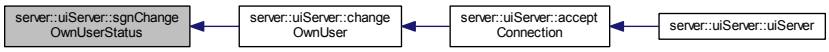

#### 7.36.3.9 void server::uiServer::sgnChangeOwnUserStatus ( QString *newStatus* ) [signal]

Emitted when a client wants to change our own status.

**Parameters**

| | |
|---|---|
| *newStatus* | The new status to take |

Here is the caller graph for this function:




#### 7.36.3.10 void server::uiServer::sgnChatReceived ( QString *message* ) [signal]

Emitted when a chatmessage has been send from a client.

**Parameters**

| | |
|---|---|
| *message* | The message that has been sent |

Here is the caller graph for this function:

#### 7.36.3.11 void server::uiServer::sgnSendEvents ( uiClient * *client* ) [signal]

Emitted when a client wants all the events.

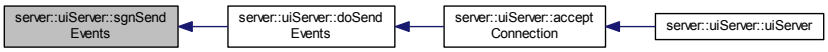
**Parameters**

| | |
|---|---|
| *client* | The client instance that wants the events |

**Note**

This is DIRECTLY hooked to the slot doSendEvents of this class

Here is the caller graph for this function:




#### 7.36.3.12 void server::uiServer::sgnUserQuery ( uiClient * *client* ) [signal]

Emitted when a client wants all the users.

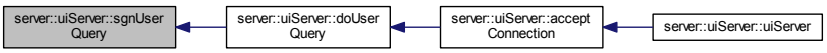
**Parameters**

| | |
|---|---|
| *client* | The client instance that wants the users |

**Note**

This is DIRECTLY hooked to the slot doUserQuery of this class

Here is the caller graph for this function:

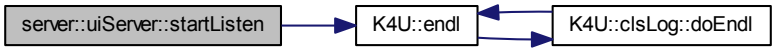



#### 7.36.3.13 void server::uiServer::startListen ( )

Starts to listen on the port.

Definition at line 18 of file uiserver.cpp.

Here is the call graph for this function:




### 7.36.4 Member Data Documentation

#### 7.36.4.1 QList<uiClient*> server::uiServer::connections [private]

Definition at line 117 of file uiserver.h.

#### 7.36.4.2 clsLog server::uiServer::log [private]

Definition at line 119 of file uiserver.h.

#### 7.36.4.3 int server::uiServer::port [private]

Definition at line 121 of file uiserver.h.

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

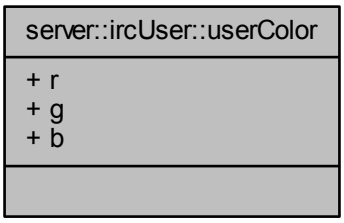
- C++/customIrcServer/uiserver.h
- C++/customIrcServer/uiserver.cpp

## 7.37 server::ircUser::userColor Struct Reference

The userColor struct As a replacement for the QColor type.

`#include <ircuser.h>`

Collaboration diagram for server::ircUser::userColor:




### Public Attributes

- int r
- int g
- int b

### 7.37.1 Detailed Description

The userColor struct As a replacement for the QColor type.

Definition at line 26 of file ircuser.h.

### 7.37.2 Member Data Documentation

#### 7.37.2.1 int server::ircUser::userColor::b

Definition at line 29 of file ircuser.h.

#### 7.37.2.2 int server::ircUser::userColor::g

Definition at line 28 of file ircuser.h.

#### 7.37.2.3 int server::ircUser::userColor::r

Definition at line 27 of file ircuser.h.

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

- C++/customIrcServer/ircuser.h

# Chapter 8

# File Documentation

## 8.1 C++/customIrcClient/android/gen/org/k4u/customIrcClient/BuildConfig.java File Reference

### Classes

- class org.k4u.customIrcClient.BuildConfig

### Packages

- package org.k4u.customIrcClient

## 8.2 BuildConfig.java

```

package org.k4u.customIrcClient;

public final class BuildConfig {
    public final static boolean DEBUG = true;
}
```

## 8.3 C++/customIrcClient/android/gen/org/k4u/customIrcClient/R.java File Reference

### Classes

- class org.k4u.customIrcClient.R
- class **org.k4u.customIrcClient.R.array**
- class **org.k4u.customIrcClient.R.attr**
- class **org.k4u.customIrcClient.R.drawable**
- class **org.k4u.customIrcClient.R.layout**
- class **org.k4u.customIrcClient.R.string**

### Packages

- package org.k4u.customIrcClient

## 8.4 R.java

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
 *
 * This class was automatically generated by the
 * aapt tool from the resource data it found.  It
 * should not be modified by hand.
 */

package org.k4u.customIrcClient;

public final class R {
    public static final class array {
        public static final int bundled_libs=0x7f040000;
        public static final int qt_libs=0x7f040001;
    }
    public static final class attr {
    }
    public static final class drawable {
        public static final int icon=0x7f020000;
        public static final int logo=0x7f020001;
    }
    public static final class layout {
        public static final int splash=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f050000;
        public static final int fatal_error_msg=0x7f050003;
        public static final int ministro_needed_msg=0x7f050002;
        public static final int ministro_not_found_msg=0x7f050001;
    }
}
```

## 8.5 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java File Reference

### Classes

- interface org.kde.necessitas.ministro.IMinistro
- class **org.kde.necessitas.ministro.IMinistro.Stub**
- class **org.kde.necessitas.ministro.IMinistro.Stub.Proxy**

### Packages

- package org.kde.necessitas.ministro

## 8.6 IMinistro.java

```
/*
 * This file is auto-generated.  DO NOT MODIFY.
 * Original file:
 E:\\repositories\\git\\CustomIRC\\C++\\customIrcClient\\android\\src\\org\\kde\\necessitas\\ministro\\IMinistro.aidl
 */
package org.kde.necessitas.ministro;
public interface IMinistro extends android.os.IInterface
{
public static abstract class Stub extends android.os.Binder implements org.kde.necessitas.ministro.
IMinistro
{
private static final java.lang.String DESCRIPTOR = "org.kde.necessitas.ministro.IMinistro";
public Stub()
{
this.attachInterface(this, DESCRIPTOR);
}
public static org.kde.necessitas.ministro.IMinistro asInterface(android.os.IBinder obj)
{
if ((obj==null)) {
return null;
}
android.os.IInterface iin = obj.queryLocalInterface(DESCRIPTOR);
if (((iin!=null)&&(iin instanceof org.kde.necessitas.ministro.IMinistro))) {
```

```
return ((org.kde.necessitas.ministro.IMinistro)iin);
}
return new org.kde.necessitas.ministro.IMinistro.Stub.Proxy(obj);
}
@Override public android.os.IBinder asBinder()
{
return this;
}
@Override public boolean onTransact(int code, android.os.Parcel data, android.os.Parcel reply, int flags)
throws android.os.RemoteException
{
switch (code)
{
case INTERFACE_TRANSACTION:
{
reply.writeString(DESCRIPTOR);
return true;
}
case TRANSACTION_requestLoader:
{
data.enforceInterface(DESCRIPTOR);
org.kde.necessitas.ministro.IMinistroCallback _arg0;
_arg0 = org.kde.necessitas.ministro.IMinistroCallback.Stub.asInterface(data.readStrongBinder());
android.os.Bundle _arg1;
if ((0!=data.readInt())) {
_arg1 = android.os.Bundle.CREATOR.createFromParcel(data);
}
else {
_arg1 = null;
}
this.requestLoader(_arg0, _arg1);
reply.writeNoException();
return true;
}
}
return super.onTransact(code, data, reply, flags);
}
private static class Proxy implements org.kde.necessitas.ministro.IMinistro
{
private android.os.IBinder mRemote;
Proxy(android.os.IBinder remote)
{
mRemote = remote;
}
@Override public android.os.IBinder asBinder()
{
return mRemote;
}
public java.lang.String getInterfaceDescriptor()
{
return DESCRIPTOR;
}
@Override public void requestLoader(org.kde.necessitas.ministro.IMinistroCallback callback,
android.os.Bundle parameters) throws android.os.RemoteException
{
android.os.Parcel _data = android.os.Parcel.obtain();
android.os.Parcel _reply = android.os.Parcel.obtain();
try {
_data.writeInterfaceToken(DESCRIPTOR);
_data.writeStrongBinder((((callback!=null))?(callback.asBinder()):(null)));
if ((parameters!=null)) {
_data.writeInt(1);
parameters.writeToParcel(_data, 0);
}
else {
_data.writeInt(0);
}
mRemote.transact(Stub.TRANSACTION_requestLoader, _data, _reply, 0);
_reply.readException();
}
finally {
_reply.recycle();
_data.recycle();
}
}
}
static final int TRANSACTION_requestLoader = (android.os.IBinder.FIRST_CALL_TRANSACTION + 0);
}
public void requestLoader(org.kde.necessitas.ministro.IMinistroCallback callback, android.os.
Bundle parameters) throws android.os.RemoteException;
}
```

## 8.7 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistro.java.d File Reference

## 8.8 IMinistro.java.d

```
E:\repositories\git\CustomIRC\C++\customIrcClient\android\gen\org\kde\necessitas\ministro\IMinistro.java: \
  E:\repositories\git\CustomIRC\C++\customIrcClient\android\src\org\kde\necessitas\ministro\IMinistro.aidl
  \
E:\repositories\git\CustomIRC\C++
  \customIrcClient\android\src\org\kde\necessitas\ministro\IMinistroCallback.aidl
```

## 8.9 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java File Reference

### Classes

- interface org.kde.necessitas.ministro.IMinistroCallback
- class **org.kde.necessitas.ministro.IMinistroCallback.Stub**
- class **org.kde.necessitas.ministro.IMinistroCallback.Stub.Proxy**

### Packages

- package org.kde.necessitas.ministro

## 8.10 IMinistroCallback.java

```
/*
 * This file is auto-generated.  DO NOT MODIFY.
 * Original file:
 E:\\repositories\\git\\CustomIRC\\C++\\customIrcClient\\android\\src\\org\\kde\\necessitas\\ministro\\IMinistroCallback.aid
 */
package org.kde.necessitas.ministro;
public interface IMinistroCallback extends android.os.IInterface
{
public static abstract class Stub extends android.os.Binder implements org.kde.necessitas.ministro.
IMinistroCallback
{
private static final java.lang.String DESCRIPTOR = "org.kde.necessitas.ministro.IMinistroCallback";
public Stub()
{
this.attachInterface(this, DESCRIPTOR);
}
public static org.kde.necessitas.ministro.IMinistroCallback asInterface(android.os.IBinder obj)
{
if ((obj==null)) {
return null;
}
android.os.IInterface iin = obj.queryLocalInterface(DESCRIPTOR);
if (((iin!=null)&&(iin instanceof org.kde.necessitas.ministro.IMinistroCallback))) {
return ((org.kde.necessitas.ministro.IMinistroCallback)iin);
}
return new org.kde.necessitas.ministro.IMinistroCallback.Stub.Proxy(obj);
}
@Override public android.os.IBinder asBinder()
{
return this;
}
@Override public boolean onTransact(int code, android.os.Parcel data, android.os.Parcel reply, int flags)
throws android.os.RemoteException
{
switch (code)
{
case INTERFACE_TRANSACTION:
{
reply.writeString(DESCRIPTOR);
```

```
return true;
}
case TRANSACTION_loaderReady:
{
data.enforceInterface(DESCRIPTOR);
android.os.Bundle _arg0;
if ((0!=data.readInt())) {
_arg0 = android.os.Bundle.CREATOR.createFromParcel(data);
}
else {
_arg0 = null;
}
this.loaderReady(_arg0);
return true;
}
}
return super.onTransact(code, data, reply, flags);
}
private static class Proxy implements org.kde.necessitas.ministro.
IMinistroCallback
{
private android.os.IBinder mRemote;
Proxy(android.os.IBinder remote)
{
mRemote = remote;
}
@Override public android.os.IBinder asBinder()
{
return mRemote;
}
public java.lang.String getInterfaceDescriptor()
{
return DESCRIPTOR;
}
@Override public void loaderReady(android.os.Bundle loaderParams) throws android.os.
RemoteException
{
android.os.Parcel _data = android.os.Parcel.obtain();
try {
_data.writeInterfaceToken(DESCRIPTOR);
if ((loaderParams!=null)) {
_data.writeInt(1);
loaderParams.writeToParcel(_data, 0);
}
else {
_data.writeInt(0);
}
mRemote.transact(Stub.TRANSACTION_loaderReady, _data, null, android.os.IBinder.FLAG_ONEWAY);
}
finally {
_data.recycle();
}
}
}
static final int TRANSACTION_loaderReady = (android.os.IBinder.FIRST_CALL_TRANSACTION + 0);
}
public void loaderReady(android.os.Bundle loaderParams) throws android.os.RemoteException;
}
```

## 8.11 C++/customIrcClient/android/gen/org/kde/necessitas/ministro/IMinistroCallback.java.d File Reference

## 8.12 IMinistroCallback.java.d

```
E:\repositories\git\CustomIRC\C++\customIrcClient\android\gen\org\kde\necessitas\ministro\IMinistroCallback
.java: \
  E:\repositories\git\CustomIRC\C++
\customIrcClient\android\src\org\kde\necessitas\ministro\IMinistroCallback.aidl
```

## 8.13 C++/customIrcClient/android/gen/R.java.d File Reference

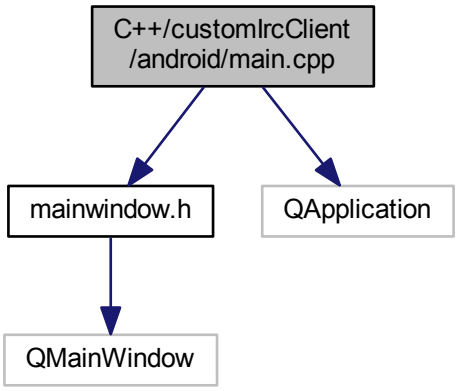
## 8.14 R.java.d

```
E:\repositories\git\CustomIRC\C++\customIrcClient\android\gen\org\k4u\customIrcClient\R.java \
 : E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\drawable\icon.png \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\drawable\logo.png \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\drawable-hdpi\icon.png \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\drawable-ldpi\icon.png \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\drawable-mdpi\icon.png \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\layout\splash.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values\libs.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-de\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-el\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-es\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-et\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-fa\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-fr\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-id\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-it\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-ja\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-ms\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-nb\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-nl\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-pl\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-pt-rBR\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-ro\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-rs\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-ru\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-zh-rCN\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\res\values-zh-rTW\strings.xml \
E:\repositories\git\CustomIRC\C++\customIrcClient\android\bin\AndroidManifest.xml \
```

## 8.15 C++/customIrcClient/android/main.cpp File Reference

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

Include dependency graph for main.cpp:




### Functions

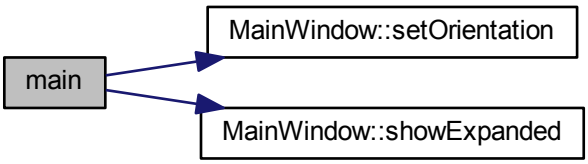
- int main (int argc, char *argv[])

### 8.15.1 Function Documentation

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

Definition at line 5 of file main.cpp.

Here is the call graph for this function:



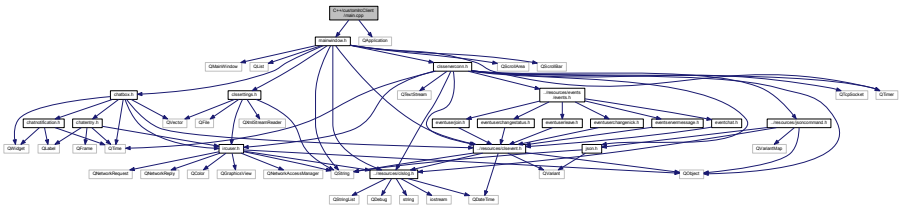

## 8.16 main.cpp

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    MainWindow mainWindow;
    mainWindow.setOrientation(MainWindow::ScreenOrientationAuto);
    mainWindow.showExpanded();

    return app.exec();
}
```

## 8.17 C++/customIrcClient/main.cpp File Reference

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

Include dependency graph for main.cpp:



### Functions

- int main (int argc, char ∗argv[])

### 8.17.1 Function Documentation

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

Definition at line 5 of file main.cpp.

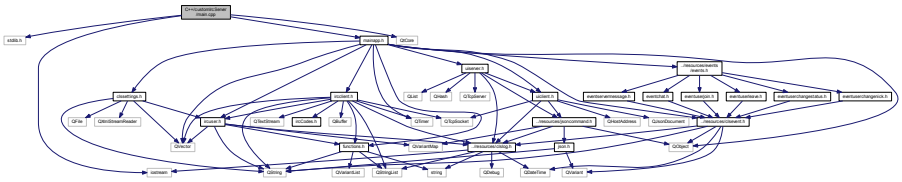
## 8.18 main.cpp

```
00001 #include "mainwindow.h"
00002 #include <QApplication>
00003
00004 using namespace client;
00005 int main(int argc, char *argv[])
00006 {
00007     QApplication a(argc, argv);
00008     MainWindow w(&a);
00009 #ifdef DESKTOP
00010     w.show();
00011 #elif ANDROID
00012     w.setOrientation(MainWindow::ScreenOrientationAuto);
00013     w.showExpanded();
00014 #endif
00015
00016     return a.exec();
00017 }
```

## 8.19 C++/customIrcServer/main.cpp File Reference

```
#include <stdlib.h>
#include <iostream>
#include <QtCore>
#include "mainapp.h"
```

Include dependency graph for main.cpp:



### Functions

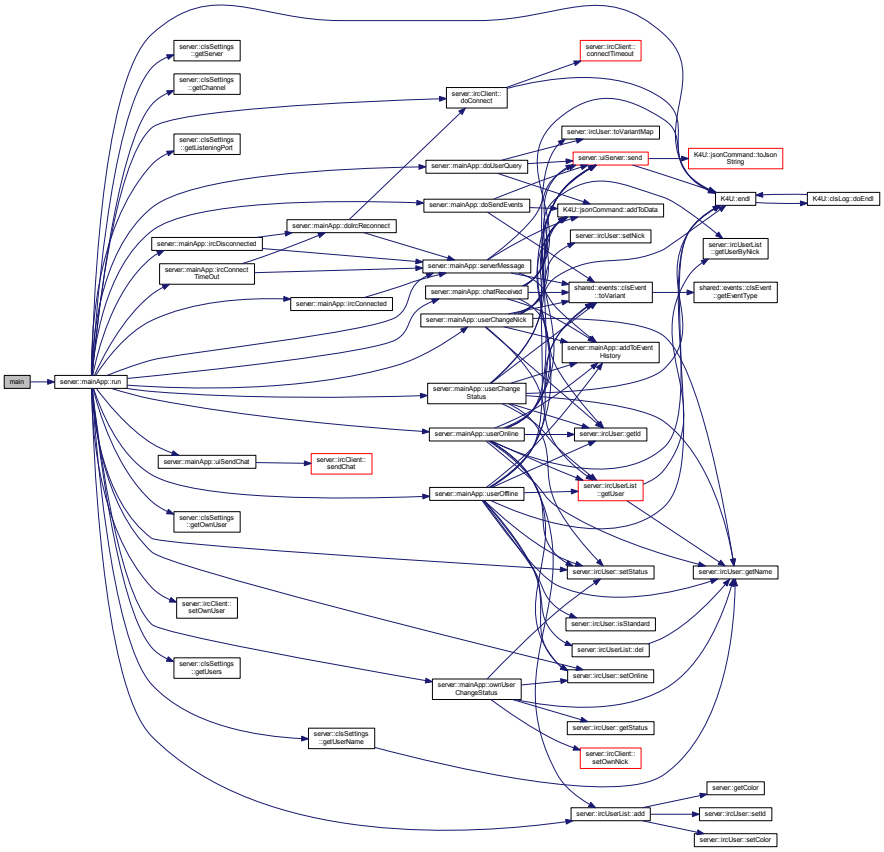
- int main (int argc, char ∗argv[])

### 8.19.1 Function Documentation

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

Definition at line 9 of file main.cpp.

Here is the call graph for this function:



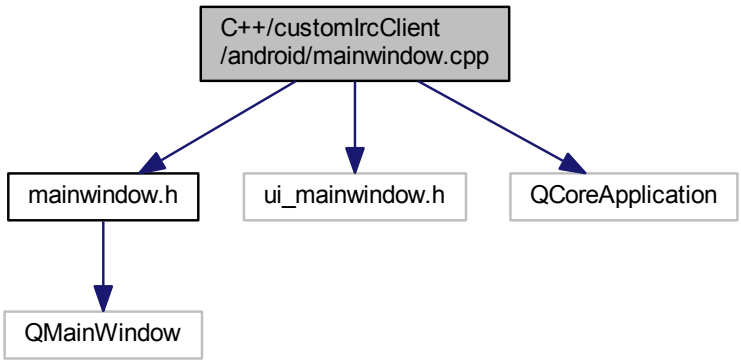

## 8.20 main.cpp

```
00001 // main.cpp
00002 #include <stdlib.h>
00003 #include <iostream>
00004 #include <QtCore>
00005 #include "mainapp.h"
00006
00007 using namespace server;
00008
00009 int main(int argc, char *argv[]){
00010     QCoreApplication a(argc, argv);
00011
00012     mainApp mApp(a.applicationDirPath(),&a);
00013
00014
00015     mApp.run();
00016     return a.exec();
00017 }
```

## 8.21 C++/customIrcClient/android/mainwindow.cpp File Reference

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QCoreApplication>
```

Include dependency graph for mainwindow.cpp:




## 8.22 mainwindow.cpp

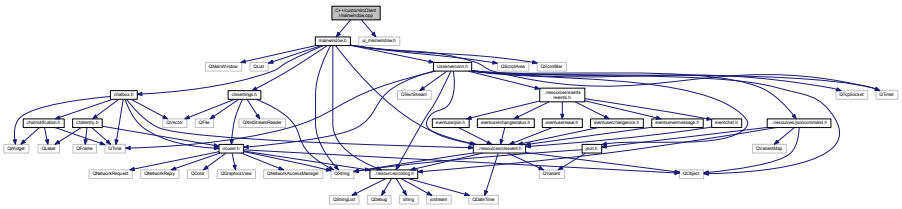
```
#include "mainwindow.h"
#include "ui_mainwindow.h"

#include <QCoreApplication>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent), ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::setOrientation(ScreenOrientation orientation)
{
    Qt::WidgetAttribute attribute;
    switch (orientation) {
#if QT_VERSION < 0x040702
    // Qt < 4.7.2 does not yet have the Qt::WA_*Orientation attributes
    case ScreenOrientationLockPortrait:
        attribute = static_cast<Qt::WidgetAttribute>(128);
        break;
    case ScreenOrientationLockLandscape:
        attribute = static_cast<Qt::WidgetAttribute>(129);
        break;
    default:
    case ScreenOrientationAuto:
        attribute = static_cast<Qt::WidgetAttribute>(130);
        break;
#else // QT_VERSION < 0x040702
    case ScreenOrientationLockPortrait:
        attribute = Qt::WA_LockPortraitOrientation;
        break;
    case ScreenOrientationLockLandscape:
        attribute = Qt::WA_LockLandscapeOrientation;
        break;
    default:
    case ScreenOrientationAuto:
        attribute = Qt::WA_AutoOrientation;
        break;
#endif // QT_VERSION < 0x040702
    };
    setAttribute(attribute, true);
}

void MainWindow::showExpanded()
{
```

```
#if defined(Q_WS_SIMULATOR)
    showFullScreen();
#elif defined(Q_WS_MAEMO_5)
    showMaximized();
#else
    show();
#endif
}
```

## 8.23 C++/customIrcClient/mainwindow.cpp File Reference

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
```

Include dependency graph for mainwindow.cpp:



### Namespaces

- namespace client

## 8.24 mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
namespace client{

    MainWindow::MainWindow(QApplication *a, QWidget *parent) :
        QMainWindow(parent),
        ui(new Ui::MainWindow),
        log(LOGTAGS_UI),
        server()
    {
        this->settings = new clsSettings(a->applicationDirPath());
        this->server = new clsServerConn(settings->getServer(),settings->getPort());
        //this->server = new clsServerConn("localhost",1337);

        ui->setupUi(this);

        connect(this->ui->scrollArea->verticalScrollBar(),SIGNAL(rangeChanged(int,int)),
                this,SLOT(moveScrollBarToBottom(int,int)));


        connect(this->server, SIGNAL(chatReceived(eventChat*)),
                this, SLOT(chatReceived(eventChat*)));
        connect(this->ui->mChatInput,SIGNAL(sendChat(QString))
                ,this,SLOT(sendChat(QString)));
        connect(this->ui->mChatInput, SIGNAL(commandGiven(QString,QList<QString>)),
                this, SLOT(commandGiven(QString,QList<QString>)));


        connect(this->server, SIGNAL(connected()), this, SLOT(serverConnected()));
        connect(this->server, SIGNAL(disconnected()), this, SLOT(serverDisconnected()));

        connect(this->server, SIGNAL(userQueryCompleted(QVector<ircUser*>)),
                this, SLOT(userQueryCompleted(QVector<ircUser*>)));

        connect(this->server, SIGNAL(userStatusChange(eventUserChangeStatus*)),
                this, SLOT(userStatusChange(
```

```
eventUserChangeStatus*)));
        connect(this->server, SIGNAL(userLeave(eventUserLeave*)),
                this, SLOT(userLeave(eventUserLeave*)));
        connect(this->server, SIGNAL(userEnter(eventUserJoin*)),
                this, SLOT(userEnter(eventUserJoin*)));
        connect(this->server, SIGNAL(userChangeNick(eventUserChangeNick*)),
                this, SLOT(userChangeNick(eventUserChangeNick*)));

        connect(this->server, SIGNAL(serverMessageReceived(eventServerMessage*)),
                this, SLOT(serverMessageReceived(eventServerMessage*)));

        connect(this->server, SIGNAL(sgnConnectTimeout()), this, SLOT(connectedTimeout()));

        this->showMessage("Connecting to server", SERVERMESSAGE_LEVEL_INFO);
        this->server->doConnect();
    }


    MainWindow::~MainWindow()
    {
        delete this->server;
        delete ui;
    }

    void MainWindow::chatReceived(eventChat *chatEvent){
        this->ui->chatbox->addChat(this->users[chatEvent->getUser()],
                chatEvent->getMessage(), chatEvent->getTime());
    }

    void MainWindow::sendChat(QString message){
        jsonCommand toSend(JSONCOMMAND_OWNCHAT);
        toSend.addToData("chat",message);

        this->server->sendCommand(toSend);
    }

    void MainWindow::commandGiven(QString command, QList<QString> args){
        if(command == "status"){
            jsonCommand toSend(JSONCOMMAND_CHANGEOWNUSER);
            toSend.addToData("change","STATUS");
            if(args.length() == 0){
                //Set status online
                toSend.addToData("new","");
            }else{
                toSend.addToData("new",args[0]);
            }
            this->server->sendCommand(toSend);
        }
    }

    void MainWindow::moveScrollBarToBottom(int min, int max){
        Q_UNUSED(min);
        this->ui->scrollArea->verticalScrollBar()->setValue(max);
    }

    void MainWindow::serverConnected(){
        //this->ui->chatbox->addMessage("Connected!");
        this->showMessage("Connected to server", SERVERMESSAGE_LEVEL_INFO);
        //Do user query
        jsonCommand toSend(JSONCOMMAND_USERQUERY);
        this->server->sendCommand(toSend);
    }


    void MainWindow::serverDisconnected(){
        QString msg("Disconnected from server. Reconnecting in %1 seconds");
        msg = msg.arg(qint32(RECONNECT_IN_S));
        this->showMessage(msg,SERVERMESSAGE_LEVEL_ERROR);
        //Now, let's reconnect in RECONNECT_IN_S
        QTimer::singleShot(RECONNECT_IN_S*1000, this,
                                        SLOT(doReconnect()));
    }

    void MainWindow::userQueryCompleted(QVector<ircUser *> users){

#ifdef DESKTOP
        //this->ui->userList->clear();
        if(this->users.count() > 0){
            this->ui->userList = new clsUserList(this);
        }
#endif
```

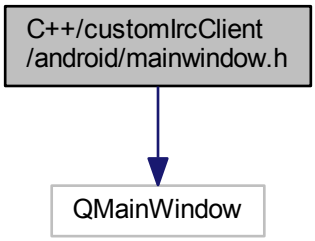
```
        this->users.clear();
        foreach(ircUser* user, users){
#ifdef DESKTOP
            this->ui->userList->addUser(user);
#endif
            this->users.insert(user->getId(),user);
        }
        //Ask for events that happened in the last time..
        jsonCommand messageQuery(JSONCOMMAND_GETEVENTS);
        this->server->sendCommand(messageQuery);
    }

    void MainWindow::userStatusChange(eventUserChangeStatus *statusEvent){
        int userId = statusEvent->getUserId();
        this->users[userId]->setStatus(statusEvent->getNewStatus());
#ifdef DESKTOP
        this->ui->userList->refreshUser(this->users[userId]);
#endif
        this->ui->chatbox->addMessage(QString(this->users[userId]->getNick()) + " is nu " +
                this->users[userId]->getStatus());
    }

    void MainWindow::userChangeNick(eventUserChangeNick *nickEvent){
        int userId = nickEvent->getUserId();
        this->users[userId]->setNick(nickEvent->getNewNick());
#ifdef DESKTOP
        this->ui->userList->refreshUser(this->users[userId]);
#endif
        this->ui->chatbox->addMessage(nickEvent->getOldNick() + " is nu bekend als " +
                this->users[userId]->getNick());
    }

    void MainWindow::userLeave(eventUserLeave *leaveEvent){
        int userId = leaveEvent->getUserId();
        QString username = "";
        if(this->users.contains(userId)){
            if(this->users[userId]->getStandard()){
                this->users[userId]->setOnline(false);
                username = this->users[userId]->getNick();
#ifdef DESKTOP
                this->ui->userList->refreshUser(this->users[userId]);
#endif
            }else{
#ifdef DESKTOP
                this->ui->userList->removeUser(this->users[userId]);
#endif
                username = this->users[userId]->getNick();
                this->users.remove(userId);
            }
            this->ui->chatbox->addMessage(username + " is weggegaan",
                                          leaveEvent->getTime());
        }
    }


    void MainWindow::userEnter(eventUserJoin *uEvent){
        ircUser *newUser = new ircUser(uEvent->getUser());
        QString username = newUser->getNick();

        if(newUser->getStandard() || this->users.keys().contains(newUser->getId())){
            //If it was already in there
            //It can only occur when bot has just started and the userquery also returns
            //a user that isn't standard
            this->users[newUser->getId()]->copyFrom(newUser);
#ifdef DESKTOP
            this->ui->userList->refreshUser(this->users[newUser->getId()]);
#endif
        }else{
            //Just add it
#ifdef DESKTOP
            this->ui->userList->addUser(newUser);
#endif
            this->users.insert(newUser->getId(),newUser);
        }
        this->ui->chatbox->addMessage(username + " is binnengekomen",
                                      uEvent->getTime());
    }

    void MainWindow::serverMessageReceived(eventServerMessage *myMessage){
        //No more, no less
        QColor colorOfMessage;
        switch(myMessage->getLevel()){
            case SERVERMESSAGE_LEVEL_ERROR:
```

```
                colorOfMessage.setRgb(160,0,0);
                break;
            case SERVERMESSAGE_LEVEL_INFO:
                colorOfMessage.setRgb(0,160,0);
                break;
        }

        QString message(myMessage->getMessage());
        this->ui->chatbox->addMessage(message, myMessage->getTime(),
                                      colorOfMessage);
    }

    void MainWindow::showMessage(QString msg, level lvl){
        QColor colorOfMessage;
        switch(lvl){
            case SERVERMESSAGE_LEVEL_ERROR:
                colorOfMessage.setRgb(160,0,0);
                break;
            case SERVERMESSAGE_LEVEL_INFO:
                colorOfMessage.setRgb(0,160,0);
                break;
        }
        this->ui->chatbox->addMessage(msg, QTime::currentTime(),
                                      colorOfMessage);
    }

    void MainWindow::connectedTimeout(){
        QString msg("Unable to connect to server. Reconnecting in %1 seconds");
        msg = msg.arg(qint32(RECONNECT_IN_S));
        this->showMessage(msg,SERVERMESSAGE_LEVEL_ERROR);
        //Now, let's reconnect in RECONNECT_IN_S
        QTimer::singleShot(RECONNECT_IN_S*1000, this,
                                       SLOT(doReconnect()));
    }

    void MainWindow::doReconnect(){
        this->showMessage("Reconnecting",SERVERMESSAGE_LEVEL_INFO);
        this->server->doConnect();
    }


#ifdef ANDROID

    void MainWindow::setOrientation(ScreenOrientation orientation)
    {
        Qt::WidgetAttribute attribute;
        switch (orientation) {
#if QT_VERSION < 0x040702
            // Qt < 4.7.2 does not yet have the Qt::WA_*Orientation attributes
            case ScreenOrientationLockPortrait:
                attribute = static_cast<Qt::WidgetAttribute>(128);
                break;
            case ScreenOrientationLockLandscape:
                attribute = static_cast<Qt::WidgetAttribute>(129);
                break;
            default:
            case ScreenOrientationAuto:
                attribute = static_cast<Qt::WidgetAttribute>(130);
                break;
#else // QT_VERSION < 0x040702
            case ScreenOrientationLockPortrait:
                attribute = Qt::WA_LockPortraitOrientation;
                break;
            case ScreenOrientationLockLandscape:
                attribute = Qt::WA_LockLandscapeOrientation;
                break;
            default:
            case ScreenOrientationAuto:
                attribute = Qt::WA_AutoOrientation;
                break;
#endif // QT_VERSION < 0x040702
        };
        setAttribute(attribute, true);
    }

    void MainWindow::showExpanded()
    {
#if defined(Q_WS_SIMULATOR)
        showFullScreen();
#elif defined(Q_WS_MAEMO_5)
        showMaximized();
#else
        show();
#endif
    }
#endif /* ANDROID */
}
```

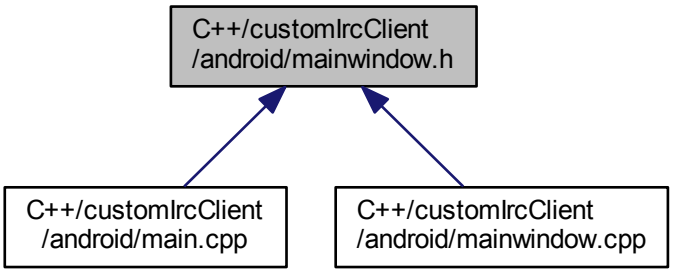
## 8.25 C++/customIrcClient/android/mainwindow.h File Reference

`#include <QMainWindow>`

Include dependency graph for mainwindow.h:




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




### Classes

- class MainWindow

### Namespaces

- namespace Ui

## 8.26 mainwindow.h

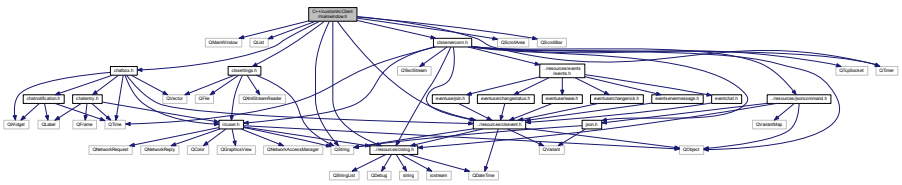
```
00001 #ifndef MAINWINDOW_H
00002 #define MAINWINDOW_H
00003
00004 #include <QMainWindow>
00005
00006 namespace Ui {
```

```
    class MainWindow;
}

class MainWindow : public QMainWindow
{
    Q_OBJECT
public:
    enum ScreenOrientation {
        ScreenOrientationLockPortrait,
        ScreenOrientationLockLandscape,
        ScreenOrientationAuto
    };

    explicit MainWindow(QWidget *parent = 0);
    virtual ~MainWindow();

    // Note that this will only have an effect on Fremantle.
    void setOrientation(ScreenOrientation orientation);

    void showExpanded();

private:
    Ui::MainWindow *ui;
};

#endif // MAINWINDOW_H
```

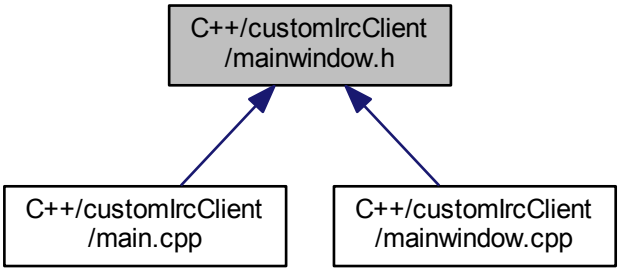
## 8.27 C++/customIrcClient/mainwindow.h File Reference

```
#include <QMainWindow>
#include <QList>
#include <QString>
#include <QScrollArea>
#include <QScrollBar>
#include <QTimer>
#include "../resources/clslog.h"
#include "../resources/clsevent.h"
#include "chatbox.h"
#include "clsserverconn.h"
#include "clssettings.h"
```

Include dependency graph for mainwindow.h:



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




### Classes

- class client::MainWindow

  *Main UI window.*

### Namespaces

- namespace Ui
- namespace client

### Macros

- #define RECONNECT_IN_S 5

## 8.27.1 Macro Definition Documentation

#### 8.27.1.1 #define RECONNECT_IN_S 5

Definition at line 21 of file mainwindow.h.

# 8.28 mainwindow.h

```
00001 #ifndef MAINWINDOW_H
00002 #define MAINWINDOW_H
00003
00004 #include <QMainWindow>
00005 #include <QList>
00006 #include <QString>
00007 #include <QScrollArea>
00008 #include <QScrollBar>
00009 #include <QTimer>
00010 #include "../resources/clslog.h"
00011 #include "../resources/clsevent.h"
00012 #include "chatbox.h"
00013 #include "clsserverconn.h"
00014 #include "clssettings.h"
00015
00016 #ifdef DESKTOP
00017 #include "desktop/clsnotify.h"
00018 #endif
```

```
#define RECONNECT_IN_S 5
namespace Ui {
    class MainWindow;
}
namespace client{

    class MainWindow : public QMainWindow
    {
        Q_OBJECT

        public:
            explicit MainWindow(QApplication *a, QWidget *parent = 0);
            ~MainWindow();

#ifdef ANDROID
            enum ScreenOrientation {
                ScreenOrientationLockPortrait,
                ScreenOrientationLockLandscape,
                ScreenOrientationAuto
            };
            // Note that this will only have an effect on Fremantle.
            void setOrientation(ScreenOrientation orientation);
            void showExpanded();
#endif


            void showMessage(QString msg, level lvl);
    private slots:
            void sendChat(QString message);
            void commandGiven(QString command, QList<QString> args);
            void moveScrollBarToBottom(int min, int max);


            void serverConnected();
            void serverDisconnected();

            void chatReceived(eventChat *chatEvent);
            void userQueryCompleted(QVector<ircUser*> users);
            void userStatusChange(eventUserChangeStatus *statusEvent);
            void userChangeNick(eventUserChangeNick* nickEvent);

            void userLeave(eventUserLeave *leaveEvent);

            void userEnter(eventUserJoin *newUser);


            void serverMessageReceived(eventServerMessage *myMessage
    );

            void connectedTimeout();

            void doReconnect();
        private:
            Ui::MainWindow *ui;

            clsLog log;
            chatBox *cb;

            clsServerConn *server;

            QMap<int, ircUser *> users;

            clsSettings *settings;
            QTimer *reconnectTimer;
    };
}
#endif // MAINWINDOW_H
```

## 8.29 C++/customIrcClient/android/src/org/kde/necessitas/origo/QtActivity.java File Reference

### Classes

- class org.kde.necessitas.origo.QtActivity

### Packages

- package org.kde.necessitas.origo

## 8.30 QtActivity.java

```
/*
    Copyright (c) 2009-2011, BogDan Vatra <bog_dan_ro@yahoo.com>
    All rights reserved.

    Redistribution and use in source and binary forms, with or without
    modification, are permitted provided that the following conditions are met:
        * Redistributions of source code must retain the above copyright
        notice, this list of conditions and the following disclaimer.
        * Redistributions in binary form must reproduce the above copyright
        notice, this list of conditions and the following disclaimer in the
        documentation and/or other materials provided with the distribution.
        * Neither the name of the  BogDan Vatra <bog_dan_ro@yahoo.com> nor the
        names of its contributors may be used to endorse or promote products
        derived from this software without specific prior written permission.

    THIS SOFTWARE IS PROVIDED BY BogDan Vatra <bog_dan_ro@yahoo.com> ''AS IS'' AND ANY
    EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
    WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
    DISCLAIMED. IN NO EVENT SHALL BogDan Vatra <bog_dan_ro@yahoo.com> BE LIABLE FOR ANY
    DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
    (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
    LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
    ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
    (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
    SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/

package org.kde.necessitas.origo;

import java.io.File;
import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.Arrays;

import org.kde.necessitas.ministro.IMinistro;
import org.kde.necessitas.ministro.IMinistroCallback;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.ComponentName;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.ServiceConnection;
import android.content.pm.ActivityInfo;
import android.content.pm.PackageManager;
import android.content.pm.PackageManager.NameNotFoundException;
import android.content.res.Configuration;
import android.content.res.Resources.Theme;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.net.Uri;
import android.os.Bundle;
import android.os.IBinder;
import android.os.RemoteException;
import android.util.AttributeSet;
import android.util.Log;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.KeyEvent;
import android.view.Menu;
import android.view.MenuItem;
import android.view.MotionEvent;
import android.view.View;
import android.view.Window;
import android.view.WindowManager.LayoutParams;
import android.view.accessibility.AccessibilityEvent;
import dalvik.system.DexClassLoader;

//@ANDROID-11
//QtCreator import android.app.Fragment;
//QtCreator import android.view.ActionMode;
//QtCreator import android.view.ActionMode.Callback;
//@ANDROID-11

```

```
public class QtActivity extends Activity
{
    private final static int MINISTRO_INSTALL_REQUEST_CODE = 0xf3ee; // request code used to know when Ministro instalation is finished
    private static final int MINISTRO_API_LEVEL=2; // Ministro api level (check IMinistro.aidl file)
    private static final int NECESSITAS_API_LEVEL=2; // Necessitas api level used by platform plugin
    private static final String QT_PROVIDER="necessitas";
    private static final int QT_VERSION=0x040801; // Qt version 4.8.00 check http://qt-project.org/doc/qt-4.8/qtglobal.html#QT_VERSION

    private static final String ERROR_CODE_KEY="error.code";
    private static final String ERROR_MESSAGE_KEY="error.message";
    private static final String DEX_PATH_KEY="dex.path";
    private static final String LIB_PATH_KEY="lib.path";
    private static final String LOADER_CLASS_NAME_KEY="loader.class.name";
    private static final String NATIVE_LIBRARIES_KEY="native.libraries";
    private static final String ENVIRONMENT_VARIABLES_KEY="environment.variables";
    private static final String APPLICATION_PARAMETERS_KEY="application.parameters";
    private static final String BUNDLED_LIBRARIES_KEY="bundled.libraries";
    private static final String MAIN_LIBRARY_KEY="main.library";
    private static final String NECESSITAS_API_LEVEL_KEY="necessitas.api.level";

    private static final String REQUIRED_MODULES_KEY="required.modules";
    private static final String APPLICATION_TITLE_KEY="application.title";
    private static final String QT_PROVIDER_KEY="qt.provider";
    private static final String MINIMUM_MINISTRO_API_KEY="minimum.ministro.api";
    private static final String MINIMUM_QT_VERSION_KEY="minimum.qt.version";
//    private static final String REPOSITORIES="3rd.party.repositories"; // needs MINISTRO_API_LEVEL >=2 !!!
                                                                        // Use this key to specify any 3rd party repositories urls
                                                                        // Ministro will download these repositories into thier
                                                                        // own folders, check http://community.kde.org/Necessitas/Ministro
                                                                        // for more details.

    private static final String APPLICATION_PARAMETERS=null; // use this variable to pass any parameters to your application,
                                                             // the parameters must not contain any white spaces
                                                             // and must be separated with "\t"
                                                             // e.g "-param1\t-param2=value2\t-param3\tvalue3"

    private static final String ENVIRONMENT_VARIABLES="QT_USE_ANDROID_NATIVE_STYLE=1\t";
                                                             // use this variable to add any environment variables to your application.
                                                             // the env vars must be separated with "\t"
                                                             // e.g. "ENV_VAR1=1\tENV_VAR2=2\t"
                                                             // Currently the following vars are used by the android plugin:
                                                             // * QT_USE_ANDROID_NATIVE_STYLE - 0 if you don't want to use android style plugin, it will save a few ms at startup.

    private static final int INCOMPATIBLE_MINISTRO_VERSION=1; // Incompatible Ministro version. Ministro needs to be upgraded.
    private ActivityInfo m_activityInfo = null; // activity info object, used to access the libs and the strings
    private DexClassLoader m_classLoader = null; // loader object
    private String[] m_qtLibs = null; // required qt libs

    // this function is used to load and start the loader
    private void loadApplication(Bundle loaderParams)
    {
        try
        {
            final int errorCode = loaderParams.getInt(ERROR_CODE_KEY);
            if (errorCode != 0)
            {
                if (errorCode == INCOMPATIBLE_MINISTRO_VERSION)
                {
                    downloadUpgradeMinistro(loaderParams.getString(ERROR_MESSAGE_KEY));
                    return;
                }

                // fatal error, show the error and quit
                AlertDialog errorDialog = new AlertDialog.Builder(QtActivity.this).create();
                errorDialog.setMessage(loaderParams.getString(ERROR_MESSAGE_KEY));
                errorDialog.setButton(getResources().getString(android.R.string.ok), new DialogInterface.OnClickListener() {
                    @Override
```

```
                    public void onClick(DialogInterface dialog, int which) {
                        finish();
                    }
                });
                errorDialog.show();
                return;
            }

            // add all bundled Qt libs to loader params
            ArrayList<String> libs = new ArrayList<String>();
            if ( m_activityInfo.metaData.containsKey("android.app.bundled_libs_resource_id") )
                libs.addAll(Arrays.asList(getResources().getStringArray(m_activityInfo.metaData.getInt("android.app.bundled_libs_resource_id"))));

            String libName = null;
            if ( m_activityInfo.metaData.containsKey("android.app.lib_name") ) {
                libName = m_activityInfo.metaData.getString("android.app.lib_name");
                loaderParams.putString(MAIN_LIBRARY_KEY, libName); //main library contains main() function
            }

            loaderParams.putStringArrayList(BUNDLED_LIBRARIES_KEY, libs);
            loaderParams.putInt(NECESSITAS_API_LEVEL_KEY, NECESSITAS_API_LEVEL);

            // load and start QtLoader class
            m_classLoader = new DexClassLoader(loaderParams.getString(DEX_PATH_KEY) // .jar/.apk files
                                               , getDir("outdex", Context.MODE_PRIVATE).getAbsolutePath() // directory where optimized DEX files should be written.
                                               , loaderParams.containsKey(LIB_PATH_KEY)?loaderParams.getString(LIB_PATH_KEY):null // libs folder (if exists)
                                               , getClassLoader()); // parent loader

            @SuppressWarnings("rawtypes")
            Class loaderClass = m_classLoader.loadClass(loaderParams.getString(LOADER_CLASS_NAME_KEY)); // load QtLoader class
            Object qtLoader = loaderClass.newInstance(); // create an instance
            Method perpareAppMethod=qtLoader.getClass().getMethod("loadApplication", Activity.class, ClassLoader.class, Bundle.class);
            if (!(Boolean)perpareAppMethod.invoke(qtLoader, this, m_classLoader, loaderParams))
                throw new Exception("");

            QtApplication.setQtActivityDelegate(qtLoader);

            // now load the application library so it's accessible from this class loader
            if (libName != null)
                System.loadLibrary(libName);

            Method startAppMethod=qtLoader.getClass().getMethod("startApplication");
            if (!(Boolean)startAppMethod.invoke(qtLoader))
                throw new Exception("");

        } catch (Exception e) {
            e.printStackTrace();
            AlertDialog errorDialog = new AlertDialog.Builder(QtActivity.this).create();
            if (m_activityInfo != null && m_activityInfo.metaData.containsKey("android.app.fatal_error_msg"))
                errorDialog.setMessage(m_activityInfo.metaData.getString("android.app.fatal_error_msg"));
            else
                errorDialog.setMessage("Fatal error, your application can't be started.");
            errorDialog.setButton(getResources().getString(android.R.string.ok), new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    finish();
                }
            });
            errorDialog.show();
        }
    }

    private ServiceConnection m_ministroConnection=new ServiceConnection() {
        private IMinistro m_service = null;
    @Override
        public void onServiceConnected(ComponentName name, IBinder service)
        {
            m_service = IMinistro.Stub.asInterface(service);
            try {
                if (m_service!=null)
                {
                    Bundle parameters= new Bundle();
                    parameters.putStringArray(REQUIRED_MODULES_KEY, m_qtLibs);
```

```
                    parameters.putString(APPLICATION_TITLE_KEY, (String) QtActivity.this.getTitle());
                    parameters.putInt(MINIMUM_MINISTRO_API_KEY, MINISTRO_API_LEVEL);
                    parameters.putString(QT_PROVIDER_KEY, QT_PROVIDER);
                    parameters.putInt(MINIMUM_QT_VERSION_KEY, QT_VERSION);
                    parameters.putString(ENVIRONMENT_VARIABLES_KEY, ENVIRONMENT_VARIABLES);
                    if (null!=APPLICATION_PARAMETERS)
                        parameters.putString(APPLICATION_PARAMETERS_KEY, APPLICATION_PARAMETERS);
                    // parameters.putStringArray(REPOSITORIES, null);
                    m_service.requestLoader(m_ministroCallback, parameters);
                }
            } catch (RemoteException e) {
                    e.printStackTrace();
            }
        }

    private IMinistroCallback m_ministroCallback = new IMinistroCallback.Stub() {
        // this function is called back by Ministro.
        @Override
        public void loaderReady(final Bundle loaderParams) throws RemoteException
        {
            runOnUiThread( new Runnable() {
                @Override
                public void run() {
                    unbindService(m_ministroConnection);
                    loadApplication(loaderParams);
                }
            });
        }
    };

        @Override
        public void onServiceDisconnected(ComponentName name)
        {
            m_service = null;
        }
    };

    private void downloadUpgradeMinistro(String msg)
    {
        AlertDialog.Builder downloadDialog = new AlertDialog.Builder(this);
        downloadDialog.setMessage(msg);
        downloadDialog.setPositiveButton(android.R.string.yes, new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                try
                {
                    Uri uri = Uri.parse("market://search?q=pname:org.kde.necessitas.ministro");
                    Intent intent = new Intent(Intent.ACTION_VIEW, uri);
                    startActivityForResult(intent, MINISTRO_INSTALL_REQUEST_CODE);
                }
                catch (Exception e) {
                    e.printStackTrace();
                    ministroNotFound();
                }
            }
        });

        downloadDialog.setNegativeButton(android.R.string.no, new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                QtActivity.this.finish();
            }
        });
        downloadDialog.show();
    }

    private void ministroNotFound()
    {
        AlertDialog errorDialog = new AlertDialog.Builder(QtActivity.this).create();

        if (m_activityInfo != null && m_activityInfo.metaData.containsKey("android.app.ministro_not_found_msg"))
            errorDialog.setMessage(m_activityInfo.metaData.getString("android.app.ministro_not_found_msg"));
        else
            errorDialog.setMessage("Can't find Ministro service.\nThe application can't start.");

        errorDialog.setButton(getResources().getString(android.R.string.ok), new DialogInterface.OnClickListener() {
```

```
            @Override
            public void onClick(DialogInterface dialog, int which) {
                finish();
            }
        });
        errorDialog.show();
    }

    private void startApp(final boolean firstStart)
    {
        try
        {
            ActivityInfo ai=getPackageManager().getActivityInfo(getComponentName(), PackageManager.GET_META_DATA);
            if (ai.metaData.containsKey("android.app.qt_libs_resource_id"))
            {
                int resourceId = ai.metaData.getInt("android.app.qt_libs_resource_id");
                m_qtLibs=getResources().getStringArray(resourceId);
            }
            if (getIntent().getExtras()!= null && getIntent().getExtras().containsKey("use_local_qt_libs")
                    && getIntent().getExtras().getString("use_local_qt_libs").equals("true"))
            {
                ArrayList<String> libraryList= new ArrayList<String>();

                String localPrefix="/data/local/qt/";
                if (getIntent().getExtras().containsKey("libs_prefix"))
                    localPrefix=getIntent().getExtras().getString("libs_prefix");

                if (m_qtLibs != null)
                    for(int i=0;i<m_qtLibs.length;i++)
                    {
                        libraryList.add(localPrefix+"lib/lib"+m_qtLibs[i]+".so");
                    }

                if (getIntent().getExtras().containsKey("load_local_libs"))
                {
                    String []extraLibs=getIntent().getExtras().getString("load_local_libs").split(":");
                    for (String lib:extraLibs)
                        if (lib.length()>0)
                            libraryList.add(localPrefix+lib);
                }

                String dexPaths = new String();
                String pathSeparator = System.getProperty("path.separator", ":");
                if (getIntent().getExtras().containsKey("load_local_jars"))
                {
                    String []jarFiles=getIntent().getExtras().getString("load_local_jars").split(":");
                    for (String jar:jarFiles)
                        if (jar.length()>0)
                        {
                            if (dexPaths.length()>0)
                                dexPaths+=pathSeparator;
                            dexPaths+=localPrefix+jar;
                        }
                }

                Bundle loaderParams = new Bundle();
                loaderParams.putInt(ERROR_CODE_KEY, 0);
                loaderParams.putString(DEX_PATH_KEY, dexPaths);
                loaderParams.putString(LOADER_CLASS_NAME_KEY, getIntent().getExtras().containsKey("loader_class_name")
                                                              ?getIntent().getExtras().getString("loader_class_name")
                                                              :"org.kde.necessitas.industrius.QtActivityDelegate");
                loaderParams.putStringArrayList(NATIVE_LIBRARIES_KEY, libraryList);
                loaderParams.putString(ENVIRONMENT_VARIABLES_KEY,"QML_IMPORT_PATH="+localPrefix+"/imports\tQT_PLUGIN_PATH="+localPrefix+"/plugins");
                loaderParams.putString(APPLICATION_PARAMETERS_KEY,"-platform\t android");
                loadApplication(loaderParams);
                return;
            }

            try {
                if (!bindService(new Intent(org.kde.necessitas.ministro.IMinistro.class.getCanonicalName()), m_ministroConnection, Context.BIND_AUTO_CREATE))
                    throw new SecurityException("");
            } catch (Exception e) {
                if (firstStart)
                {
                    String msg="This application requires Ministro service. Would you like to install it?";
                    if (m_activityInfo != null && m_activityInfo.metaData.containsKey("android.app.ministro_needed_msg"))
                        msg=m_activityInfo.metaData.getString("android.app.ministro_needed_msg");
                    downloadUpgradeMinistro(msg);
```

```
                }
                else
                {
                    ministroNotFound();
                }
            }
        }
        catch (Exception e)
        {
            Log.e(QtApplication.QtTAG, "Can't create main activity", e);
        }
    }



    @Override
    public boolean dispatchKeyEvent(KeyEvent event)
    {
        if (QtApplication.m_delegateObject != null &&
QtApplication.dispatchKeyEvent != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.dispatchKeyEvent, event);
        else
            return super.dispatchKeyEvent(event);
    }
    public boolean super_dispatchKeyEvent(KeyEvent event)
    {
        return super.dispatchKeyEvent(event);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean dispatchPopulateAccessibilityEvent(AccessibilityEvent
event)
    {
        if (QtApplication.m_delegateObject != null &&
QtApplication.dispatchPopulateAccessibilityEvent != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.dispatchPopulateAccessibilityEvent, event);
        else
            return super.dispatchPopulateAccessibilityEvent(event);
    }
    public boolean super_dispatchPopulateAccessibilityEvent(
AccessibilityEvent event)
    {
        return super_dispatchPopulateAccessibilityEvent(event);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean dispatchTouchEvent(MotionEvent ev)
    {
        if (QtApplication.m_delegateObject != null &&
QtApplication.dispatchTouchEvent != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.dispatchTouchEvent, ev);
        else
            return super.dispatchTouchEvent(ev);
    }
    public boolean super_dispatchTouchEvent(MotionEvent event)
    {
        return super.dispatchTouchEvent(event);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean dispatchTrackballEvent(MotionEvent ev)
    {
        if (QtApplication.m_delegateObject != null &&
QtApplication.dispatchTrackballEvent != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.dispatchTrackballEvent, ev);
        else
            return super.dispatchTrackballEvent(ev);
    }
    public boolean super_dispatchTrackballEvent(MotionEvent event)
    {
        return super.dispatchTrackballEvent(event);
    }
    //----------------------------------------------------------------------------

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data)
    {

        if (QtApplication.m_delegateObject != null &&
```

```
QtApplication.onActivityResult != null)
        {
            QtApplication.invokeDelegateMethod(QtApplication.onActivityResult, requestCode, resultCode, data);
            return;
        }
        if (requestCode == MINISTRO_INSTALL_REQUEST_CODE)
                startApp(false);
        super.onActivityResult(requestCode, resultCode, data);
    }
    public void super_onActivityResult(int requestCode, int resultCode, Intent data)
    {
        super.onActivityResult(requestCode, resultCode, data);
    }
    //----------------------------------------------------------------------------

    @Override
    protected void onApplyThemeResource(Theme theme, int resid, boolean first)
    {
        if (!QtApplication.invokeDelegate(theme, resid, first).invoked)
            super.onApplyThemeResource(theme, resid, first);
    }
    public void super_onApplyThemeResource(Theme theme, int resid, boolean first)
    {
        super.onApplyThemeResource(theme, resid, first);
    }
    //----------------------------------------------------------------------------


    @Override
    protected void onChildTitleChanged(Activity childActivity, CharSequence title)
    {
        if (!QtApplication.invokeDelegate(childActivity, title).invoked)
            super.onChildTitleChanged(childActivity, title);
    }
    public void super_onChildTitleChanged(Activity childActivity, CharSequence title)
    {
        super.onChildTitleChanged(childActivity, title);
    }
    //----------------------------------------------------------------------------

    @Override
    public void onConfigurationChanged(Configuration newConfig)
    {
        if (!QtApplication.invokeDelegate(newConfig).invoked)
            super.onConfigurationChanged(newConfig);
    }
    public void super_onConfigurationChanged(Configuration newConfig)
    {
        super.onConfigurationChanged(newConfig);
    }
    //----------------------------------------------------------------------------

    @Override
    public void onContentChanged()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onContentChanged();
    }
    public void super_onContentChanged()
    {
        super.onContentChanged();
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onContextItemSelected(MenuItem item)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(item);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onContextItemSelected(item);
    }
    public boolean super_onContextItemSelected(MenuItem item)
    {
        return super.onContextItemSelected(item);
    }
    //----------------------------------------------------------------------------

    @Override
    public void onContextMenuClosed(Menu menu)
    {
        if (!QtApplication.invokeDelegate(menu).invoked)
            super.onContextMenuClosed(menu);
```

```
    }
    public void super_onContextMenuClosed(Menu menu)
    {
        super.onContextMenuClosed(menu);
    }
    //----------------------------------------------------------------------------

    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        if (QtApplication.m_delegateObject != null &&
QtApplication.onCreate != null)
        {
            QtApplication.invokeDelegateMethod(
QtApplication.onCreate, savedInstanceState);
            return;
        }
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        try {
            m_activityInfo = getPackageManager().getActivityInfo(getComponentName(),
PackageManager.GET_META_DATA);
        } catch (NameNotFoundException e) {
            e.printStackTrace();
            finish();
            return;
        }
        if (null == getLastNonConfigurationInstance())
        {
            // if splash screen is defined, then show it
            if ( m_activityInfo.metaData.containsKey("android.app.splash_screen") )
                setContentView(m_activityInfo.metaData.getInt("android.app.splash_screen"));
            startApp(true);
        }
    }
    //----------------------------------------------------------------------------

    @Override
    public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo)
    {
        if (!QtApplication.invokeDelegate(menu, v, menuInfo).invoked)
            super.onCreateContextMenu(menu, v, menuInfo);
    }
    public void super_onCreateContextMenu(ContextMenu menu, View v,
ContextMenuInfo menuInfo)
    {
        super.onCreateContextMenu(menu, v, menuInfo);
    }
    //----------------------------------------------------------------------------

    @Override
    public CharSequence onCreateDescription()
    {
        QtApplication.InvokeResult res = QtApplication.
invokeDelegate();
        if (res.invoked)
            return (CharSequence)res.methodReturns;
        else
            return super.onCreateDescription();
    }
    public CharSequence super_onCreateDescription()
    {
        return super.onCreateDescription();
    }
    //----------------------------------------------------------------------------

    @Override
    protected Dialog onCreateDialog(int id)
    {
        QtApplication.InvokeResult res = QtApplication.
invokeDelegate(id);
        if (res.invoked)
            return (Dialog)res.methodReturns;
        else
            return super.onCreateDialog(id);
    }
    public Dialog super_onCreateDialog(int id)
    {
        return super.onCreateDialog(id);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        QtApplication.InvokeResult res = QtApplication.
invokeDelegate(menu);
```

```
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onCreateOptionsMenu(menu);
    }
    public boolean super_onCreateOptionsMenu(Menu menu)
    {
        return super.onCreateOptionsMenu(menu);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onCreatePanelMenu(int featureId, Menu menu)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(featureId, menu);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onCreatePanelMenu(featureId, menu);
    }
    public boolean super_onCreatePanelMenu(int featureId, Menu menu)
    {
        return super.onCreatePanelMenu(featureId, menu);
    }
    //----------------------------------------------------------------------------


    @Override
    public View onCreatePanelView(int featureId)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(featureId);
        if (res.invoked)
            return (View)res.methodReturns;
        else
            return super.onCreatePanelView(featureId);
    }
    public View super_onCreatePanelView(int featureId)
    {
        return super.onCreatePanelView(featureId);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onCreateThumbnail(Bitmap outBitmap, Canvas canvas)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(outBitmap, canvas);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onCreateThumbnail(outBitmap, canvas);
    }
    public boolean super_onCreateThumbnail(Bitmap outBitmap, Canvas canvas)
    {
        return super.onCreateThumbnail(outBitmap, canvas);
    }
    //----------------------------------------------------------------------------

    @Override
    public View onCreateView(String name, Context context, AttributeSet attrs)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(name, context, attrs);
        if (res.invoked)
            return (View)res.methodReturns;
        else
            return super.onCreateView(name, context, attrs);
    }
    public View super_onCreateView(String name, Context context, AttributeSet attrs)
    {
        return super.onCreateView(name, context, attrs);
    }
    //----------------------------------------------------------------------------

    @Override
    protected void onDestroy()
    {
        super.onDestroy();
        QtApplication.invokeDelegate();
    }
    //----------------------------------------------------------------------------


    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event)
```

```
    {
        if (QtApplication.m_delegateObject != null &&
QtApplication.onKeyDown != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.onKeyDown, keyCode, event);
        else
            return super.onKeyDown(keyCode, event);
    }
    public boolean super_onKeyDown(int keyCode, KeyEvent event)
    {
        return super.onKeyDown(keyCode, event);
    }
    //----------------------------------------------------------------------------


    @Override
    public boolean onKeyMultiple(int keyCode, int repeatCount, KeyEvent event)
    {
        if (QtApplication.m_delegateObject != null &&
QtApplication.onKeyMultiple != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.onKeyMultiple ,keyCode, repeatCount, event);
        else
            return super.onKeyMultiple(keyCode, repeatCount, event);
    }
    public boolean super_onKeyMultiple(int keyCode, int repeatCount, KeyEvent event)
    {
        return super.onKeyMultiple(keyCode, repeatCount, event);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onKeyUp(int keyCode, KeyEvent event)
    {
        if (QtApplication.m_delegateObject != null  &&
QtApplication.onKeyDown != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
QtApplication.onKeyUp, keyCode, event);
        else
            return super.onKeyUp(keyCode, event);
    }
    public boolean super_onKeyUp(int keyCode, KeyEvent event)
    {
        return super.onKeyUp(keyCode, event);
    }
    //----------------------------------------------------------------------------

    @Override
    public void onLowMemory()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onLowMemory();
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onMenuItemSelected(int featureId, MenuItem item)
    {
        QtApplication.InvokeResult res = QtApplication.
invokeDelegate(featureId, item);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onMenuItemSelected(featureId, item);
    }
    public boolean super_onMenuItemSelected(int featureId, MenuItem item)
    {
        return super.onMenuItemSelected(featureId, item);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onMenuOpened(int featureId, Menu menu)
    {
        QtApplication.InvokeResult res = QtApplication.
invokeDelegate(featureId, menu);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onMenuOpened(featureId, menu);
    }
    public boolean super_onMenuOpened(int featureId, Menu menu)
    {
        return super.onMenuOpened(featureId, menu);
    }
    //----------------------------------------------------------------------------

```

```
    @Override
    protected void onNewIntent(Intent intent)
    {
        if (!QtApplication.invokeDelegate(intent).invoked)
            super.onNewIntent(intent);
    }
    public void super_onNewIntent(Intent intent)
    {
        super.onNewIntent(intent);
    }
    //---------------------------------------------------------------------------

    @Override
    public boolean onOptionsItemSelected(MenuItem item)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(item);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onOptionsItemSelected(item);
    }
    public boolean super_onOptionsItemSelected(MenuItem item)
    {
        return super.onOptionsItemSelected(item);
    }
    //---------------------------------------------------------------------------

    @Override
    public void onOptionsMenuClosed(Menu menu)
    {
        if (!QtApplication.invokeDelegate(menu).invoked)
            super.onOptionsMenuClosed(menu);
    }
    public void super_onOptionsMenuClosed(Menu menu)
    {
        super.onOptionsMenuClosed(menu);
    }
    //---------------------------------------------------------------------------

    @Override
    public void onPanelClosed(int featureId, Menu menu)
    {
        if (!QtApplication.invokeDelegate(featureId, menu).invoked)
            super.onPanelClosed(featureId, menu);
    }
    public void super_onPanelClosed(int featureId, Menu menu)
    {
        super.onPanelClosed(featureId, menu);
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onPause()
    {
        super.onPause();
        QtApplication.invokeDelegate();
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onPostCreate(Bundle savedInstanceState)
    {
        super.onPostCreate(savedInstanceState);
        QtApplication.invokeDelegate(savedInstanceState);
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onPostResume()
    {
        super.onPostResume();
        QtApplication.invokeDelegate();
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onPrepareDialog(int id, Dialog dialog)
    {
        if (!QtApplication.invokeDelegate(id, dialog).invoked)
            super.onPrepareDialog(id, dialog);
    }
    public void super_onPrepareDialog(int id, Dialog dialog)
    {
        super.onPrepareDialog(id, dialog);
    }
    //---------------------------------------------------------------------------
```

```
    @Override
    public boolean onPrepareOptionsMenu(Menu menu)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(menu);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onPrepareOptionsMenu(menu);
    }
    public boolean super_onPrepareOptionsMenu(Menu menu)
    {
        return super.onPrepareOptionsMenu(menu);
    }
    //---------------------------------------------------------------------------

    @Override
    public boolean onPreparePanel(int featureId, View view, Menu menu)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(featureId, view, menu);
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onPreparePanel(featureId, view, menu);
    }
    public boolean super_onPreparePanel(int featureId, View view, Menu menu)
    {
        return super.onPreparePanel(featureId, view, menu);
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onRestart()
    {
        super.onRestart();
        QtApplication.invokeDelegate();
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState)
    {
        if (!QtApplication.invokeDelegate(savedInstanceState).invoked)
            super.onRestoreInstanceState(savedInstanceState);
    }
    public void super_onRestoreInstanceState(Bundle savedInstanceState)
    {
        super.onRestoreInstanceState(savedInstanceState);
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onResume()
    {
        super.onResume();
        QtApplication.invokeDelegate();
    }
    //---------------------------------------------------------------------------

    @Override
    public Object onRetainNonConfigurationInstance()
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate();
        if (res.invoked)
            return res.methodReturns;
        else
            return super.onRetainNonConfigurationInstance();
    }
    public Object super_onRetainNonConfigurationInstance()
    {
        return super.onRetainNonConfigurationInstance();
    }
    //---------------------------------------------------------------------------

    @Override
    protected void onSaveInstanceState(Bundle outState)
    {
        if (!QtApplication.invokeDelegate(outState).invoked)
            super.onSaveInstanceState(outState);
    }
    public void super_onSaveInstanceState(Bundle outState)
    {
        super.onSaveInstanceState(outState);

```

```
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onSearchRequested()
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate();
        if (res.invoked)
            return (Boolean)res.methodReturns;
        else
            return super.onSearchRequested();
    }
    public boolean super_onSearchRequested()
    {
        return super.onSearchRequested();
    }
    //----------------------------------------------------------------------------

    @Override
    protected void onStart()
    {
        super.onStart();
        QtApplication.invokeDelegate();
    }
    //----------------------------------------------------------------------------

    @Override
    protected void onStop()
    {
        super.onStop();
        QtApplication.invokeDelegate();
    }
    //----------------------------------------------------------------------------

    @Override
    protected void onTitleChanged(CharSequence title, int color)
    {
        if (!QtApplication.invokeDelegate(title, color).invoked)
            super.onTitleChanged(title, color);
    }
    public void super_onTitleChanged(CharSequence title, int color)
    {
        super.onTitleChanged(title, color);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onTouchEvent(MotionEvent event)
    {
        if (QtApplication.m_delegateObject != null  && QtApplication.onTouchEvent != null)
            return (Boolean) QtApplication.invokeDelegateMethod(QtApplication.onTouchEvent, event);
        else
            return super.onTouchEvent(event);
    }
    public boolean super_onTouchEvent(MotionEvent event)
    {
        return super.onTouchEvent(event);
    }
    //----------------------------------------------------------------------------

    @Override
    public boolean onTrackballEvent(MotionEvent event)
    {
        if (QtApplication.m_delegateObject != null  && QtApplication.onTrackballEvent != null)
            return (Boolean) QtApplication.invokeDelegateMethod(QtApplication.onTrackballEvent, event);
        else
            return super.onTrackballEvent(event);
    }
    public boolean super_onTrackballEvent(MotionEvent event)
    {
        return super.onTrackballEvent(event);
    }
    //----------------------------------------------------------------------------

    @Override
    public void onUserInteraction()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onUserInteraction();
    }
    public void super_onUserInteraction()
    {
```

```
        super.onUserInteraction();
    }
    //--------------------------------------------------------------------------

    @Override
    protected void onUserLeaveHint()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onUserLeaveHint();
    }
    public void super_onUserLeaveHint()
    {
        super.onUserLeaveHint();
    }
    //--------------------------------------------------------------------------

    @Override
    public void onWindowAttributesChanged(LayoutParams params)
    {
        if (!QtApplication.invokeDelegate(params).invoked)
            super.onWindowAttributesChanged(params);
    }
    public void super_onWindowAttributesChanged(LayoutParams params)
    {
        super.onWindowAttributesChanged(params);
    }
    //--------------------------------------------------------------------------

    @Override
    public void onWindowFocusChanged(boolean hasFocus)
    {
        if (!QtApplication.invokeDelegate(hasFocus).invoked)
            super.onWindowFocusChanged(hasFocus);
    }
    public void super_onWindowFocusChanged(boolean hasFocus)
    {
        super.onWindowFocusChanged(hasFocus);
    }
    //--------------------------------------------------------------------------

//@ANDROID-5
    @Override
    public void onAttachedToWindow()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onAttachedToWindow();
    }
    public void super_onAttachedToWindow()
    {
        super.onAttachedToWindow();
    }
    //--------------------------------------------------------------------------

    @Override
    public void onBackPressed()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onBackPressed();
    }
    public void super_onBackPressed()
    {
        super.onBackPressed();
    }
    //--------------------------------------------------------------------------

    @Override
    public void onDetachedFromWindow()
    {
        if (!QtApplication.invokeDelegate().invoked)
            super.onDetachedFromWindow();
    }
    public void super_onDetachedFromWindow()
    {
        super.onDetachedFromWindow();
    }
    //--------------------------------------------------------------------------

    @Override
    public boolean onKeyLongPress(int keyCode, KeyEvent event)
    {
        if (QtApplication.m_delegateObject != null  &&
    QtApplication.onKeyLongPress != null)
            return (Boolean) QtApplication.invokeDelegateMethod(
    QtApplication.onKeyLongPress, keyCode, event);
        else
            return super.onKeyLongPress(keyCode, event);
    }
```

```
    public boolean super_onKeyLongPress(int keyCode, KeyEvent event)
    {
        return super.onKeyLongPress(keyCode, event);
    }
    //--------------------------------------------------------------------------
//@ANDROID-5

//@ANDROID-8
@Override
    protected Dialog onCreateDialog(int id, Bundle args)
    {
        QtApplication.InvokeResult res = QtApplication.invokeDelegate(id, args);
        if (res.invoked)
            return (Dialog)res.methodReturns;
        else
            return super.onCreateDialog(id, args);
    }
    public Dialog super_onCreateDialog(int id, Bundle args)
    {
        return super.onCreateDialog(id, args);
    }
    //--------------------------------------------------------------------------

    @Override
    protected void onPrepareDialog(int id, Dialog dialog, Bundle args)
    {
        if (!QtApplication.invokeDelegate(id, dialog, args).invoked)
            super.onPrepareDialog(id, dialog, args);
    }
    public void super_onPrepareDialog(int id, Dialog dialog, Bundle args)
    {
        super.onPrepareDialog(id, dialog, args);
    }
    //--------------------------------------------------------------------------
//@ANDROID-8

//@ANDROID-11
//QtCreator     @Override
//QtCreator     public boolean dispatchKeyShortcutEvent(KeyEvent event)
//QtCreator     {
//QtCreator         if (QtApplication.m_delegateObject != null  && QtApplication.dispatchKeyShortcutEvent != null)
//QtCreator             return (Boolean) QtApplication.invokeDelegateMethod(QtApplication.dispatchKeyShortcutEvent, event);
//QtCreator         else
//QtCreator             return super.dispatchKeyShortcutEvent(event);
//QtCreator     }
//QtCreator     public boolean super_dispatchKeyShortcutEvent(KeyEvent event)
//QtCreator     {
//QtCreator         return super.dispatchKeyShortcutEvent(event);
//QtCreator     }
//QtCreator     //------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public void onActionModeFinished(ActionMode mode)
//QtCreator     {
//QtCreator         if (!QtApplication.invokeDelegate(mode).invoked)
//QtCreator             super.onActionModeFinished(mode);
//QtCreator     }
//QtCreator     public void super_onActionModeFinished(ActionMode mode)
//QtCreator     {
//QtCreator         super.onActionModeFinished(mode);
//QtCreator     }
//QtCreator     //------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public void onActionModeStarted(ActionMode mode)
//QtCreator     {
//QtCreator         if (!QtApplication.invokeDelegate(mode).invoked)
//QtCreator             super.onActionModeStarted(mode);
//QtCreator     }
//QtCreator     public void super_onActionModeStarted(ActionMode mode)
//QtCreator     {
//QtCreator         super.onActionModeStarted(mode);
//QtCreator     }
//QtCreator     //------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public void onAttachFragment(Fragment fragment)
//QtCreator     {
//QtCreator         if (!QtApplication.invokeDelegate(fragment).invoked)
//QtCreator             super.onAttachFragment(fragment);
//QtCreator     }
//QtCreator     public void super_onAttachFragment(Fragment fragment)
//QtCreator     {
//QtCreator         super.onAttachFragment(fragment);
```

```
//QtCreator     }
//QtCreator     //----------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public View onCreateView(View parent, String name, Context context, AttributeSet attrs)
//QtCreator     {
//QtCreator         QtApplication.InvokeResult res = QtApplication.invokeDelegate(parent, name, context, attrs);
//QtCreator         if (res.invoked)
//QtCreator             return (View)res.methodReturns;
//QtCreator         else
//QtCreator             return super.onCreateView(parent, name, context, attrs);
//QtCreator     }
//QtCreator     public View super_onCreateView(View parent, String name, Context context,
//QtCreator             AttributeSet attrs) {
//QtCreator         return super.onCreateView(parent, name, context, attrs);
//QtCreator     }
//QtCreator     //----------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public boolean onKeyShortcut(int keyCode, KeyEvent event)
//QtCreator     {
//QtCreator         if (QtApplication.m_delegateObject != null  && QtApplication.onKeyShortcut != null)
//QtCreator             return (Boolean) QtApplication.invokeDelegateMethod(QtApplication.onKeyShortcut, keyCode,event);
//QtCreator         else
//QtCreator             return super.onKeyShortcut(keyCode, event);
//QtCreator     }
//QtCreator     public boolean super_onKeyShortcut(int keyCode, KeyEvent event)
//QtCreator     {
//QtCreator         return super.onKeyShortcut(keyCode, event);
//QtCreator     }
//QtCreator     //----------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public ActionMode onWindowStartingActionMode(Callback callback)
//QtCreator     {
//QtCreator         QtApplication.InvokeResult res = QtApplication.invokeDelegate(callback);
//QtCreator         if (res.invoked)
//QtCreator             return (ActionMode)res.methodReturns;
//QtCreator         else
//QtCreator             return super.onWindowStartingActionMode(callback);
//QtCreator     }
//QtCreator     public ActionMode super_onWindowStartingActionMode(Callback callback)
//QtCreator     {
//QtCreator         return super.onWindowStartingActionMode(callback);
//QtCreator     }
//QtCreator     //----------------------------------------------------------------------------
//@ANDROID-11

//@ANDROID-12
//QtCreator     @Override
//QtCreator     public boolean dispatchGenericMotionEvent(MotionEvent ev)
//QtCreator     {
//QtCreator         if (QtApplication.m_delegateObject != null  && QtApplication.dispatchGenericMotionEvent != null)
//QtCreator             return (Boolean) QtApplication.invokeDelegateMethod(QtApplication.dispatchGenericMotionEvent, ev);
//QtCreator         else
//QtCreator             return super.dispatchGenericMotionEvent(ev);
//QtCreator     }
//QtCreator     public boolean super_dispatchGenericMotionEvent(MotionEvent event)
//QtCreator     {
//QtCreator         return super.dispatchGenericMotionEvent(event);
//QtCreator     }
//QtCreator     //----------------------------------------------------------------------------
//QtCreator
//QtCreator     @Override
//QtCreator     public boolean onGenericMotionEvent(MotionEvent event)
//QtCreator     {
//QtCreator         if (QtApplication.m_delegateObject != null  && QtApplication.onGenericMotionEvent != null)
//QtCreator             return (Boolean) QtApplication.invokeDelegateMethod(QtApplication.onGenericMotionEvent, event);
//QtCreator         else
//QtCreator             return super.onGenericMotionEvent(event);
//QtCreator     }
//QtCreator     public boolean super_onGenericMotionEvent(MotionEvent event)
//QtCreator     {
//QtCreator         return super.onGenericMotionEvent(event);
//QtCreator     }
//QtCreator     //----------------------------------------------------------------------------
//@ANDROID-12

}
```

## 8.31 C++/customIrcClient/android/src/org/kde/necessitas/origo/QtApplication.java File Reference

### Classes

- class org.kde.necessitas.origo.QtApplication
- class **org.kde.necessitas.origo.QtApplication.InvokeResult**

### Packages

- package org.kde.necessitas.origo

## 8.32 QtApplication.java

```
00001 /*
00002     Copyright (c) 2009-2011, BogDan Vatra <bog_dan_ro@yahoo.com>
00003     All rights reserved.
00004
00005     Redistribution and use in source and binary forms, with or without
00006     modification, are permitted provided that the following conditions are met:
00007         * Redistributions of source code must retain the above copyright
00008         notice, this list of conditions and the following disclaimer.
00009         * Redistributions in binary form must reproduce the above copyright
00010         notice, this list of conditions and the following disclaimer in the
00011         documentation and/or other materials provided with the distribution.
00012         * Neither the name of the  BogDan Vatra <bog_dan_ro@yahoo.com> nor the
00013         names of its contributors may be used to endorse or promote products
00014         derived from this software without specific prior written permission.
00015
00016     THIS SOFTWARE IS PROVIDED BY BogDan Vatra <bog_dan_ro@yahoo.com> ''AS IS'' AND ANY
00017     EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
00018     WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
00019     DISCLAIMED. IN NO EVENT SHALL BogDan Vatra <bog_dan_ro@yahoo.com> BE LIABLE FOR ANY
00020     DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
00021     (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
00022     LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
00023     ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
00024     (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
00025     SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00026 */
00027
00028 package org.kde.necessitas.origo;
00029
00030 import java.lang.reflect.Field;
00031 import java.lang.reflect.Method;
00032 import java.util.ArrayList;
00033 import java.util.HashMap;
00034
00035 import android.app.Application;
00036
00037 public class QtApplication extends Application
00038 {
00039     public final static String QtTAG="Qt";
00040     public static Object m_delegateObject = null;
00041     public static HashMap<String, ArrayList<Method>> m_delegateMethods= new
      HashMap<String, ArrayList<Method>>();
00042     public static Method dispatchKeyEvent = null;
00043     public static Method dispatchPopulateAccessibilityEvent = null;
00044     public static Method dispatchTouchEvent = null;
00045     public static Method dispatchTrackballEvent = null;
00046     public static Method onKeyDown = null;
00047     public static Method onKeyMultiple = null;
00048     public static Method onKeyUp = null;
00049     public static Method onTouchEvent = null;
00050     public static Method onTrackballEvent = null;
00051     public static Method onActivityResult = null;
00052     public static Method onCreate = null;
00053     public static Method onKeyLongPress = null;
00054     public static Method dispatchKeyShortcutEvent = null;
00055     public static Method onKeyShortcut = null;
00056     public static Method dispatchGenericMotionEvent = null;
00057     public static Method onGenericMotionEvent = null;
00058 /*
00059     public static Method onTerminate = null;
00060     public static Method onApplyThemeResource = null;
```

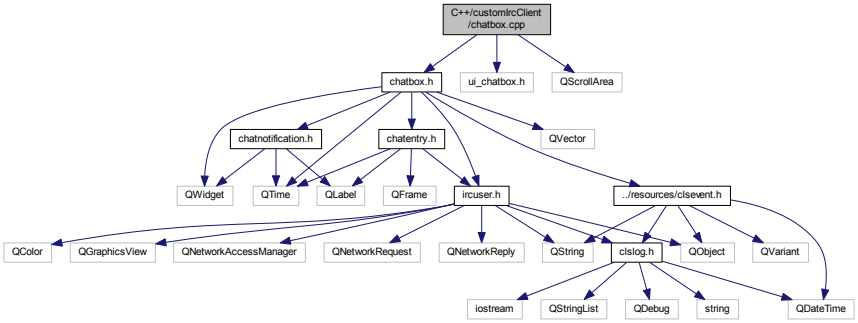
```
    public static Method onChildTitleChanged = null;
    public static Method onConfigurationChanged = null;
    public static Method onContentChanged  = null;
    public static Method onContextItemSelected  = null;
    public static Method onContextMenuClosed = null;
    public static Method onCreateContextMenu = null;
    public static Method onCreateDescription = null;
    public static Method onCreateDialog = null;
    public static Method onCreateOptionsMenu = null;
    public static Method onCreatePanelMenu = null;
    public static Method onCreatePanelView = null;
    public static Method onCreateThumbnail = null;
    public static Method onCreateView = null;
    public static Method onDestroy = null;
    public static Method onLowMemory = null;
    public static Method onMenuItemSelected = null;
    public static Method onMenuOpened = null;
    public static Method onNewIntent = null;
    public static Method onOptionsItemSelected = null;
    public static Method onOptionsMenuClosed = null;
    public static Method onPanelClosed = null;
    public static Method onPause = null;
    public static Method onPostCreate = null;
    public static Method onPostResume = null;
    public static Method onPrepareDialog = null;
    public static Method onPrepareOptionsMenu = null;
    public static Method onPreparePanel = null;
    public static Method onRestart = null;
    public static Method onRestoreInstanceState = null;
    public static Method onResume = null;
    public static Method onRetainNonConfigurationInstance = null;
    public static Method onSaveInstanceState = null;
    public static Method onSearchRequested = null;
    public static Method onStart = null;
    public static Method onStop = null;
    public static Method onTitleChanged = null;
    public static Method onUserInteraction = null;
    public static Method onUserLeaveHint = null;
    public static Method onWindowAttributesChanged = null;
    public static Method onWindowFocusChanged = null;
    public static Method onAttachedToWindow = null;
    public static Method onBackPressed = null;
    public static Method onDetachedFromWindow = null;
    public static Method onCreateDialog8 = null;
    public static Method onPrepareDialog8 = null;
    public static Method onActionModeFinished = null;
    public static Method onActionModeStarted = null;
    public static Method onAttachFragment = null;
    public static Method onCreateView11 = null;
    public static Method onWindowStartingActionMode = null;
*/

    public static void setQtActivityDelegate(Object listener)
    {
        QtApplication.m_delegateObject = listener;

        ArrayList<Method> delegateMethods = new ArrayList<Method>();
        for (Method m: listener.getClass().getMethods())
            if (m.getDeclaringClass().getName().startsWith("org.kde.necessitas"))
                delegateMethods.add(m);

        ArrayList<Field> applicationFields = new ArrayList<Field>();
        for (Field f: QtApplication.class.getFields())
            if (f.getDeclaringClass().getName().equals(QtApplication.class.getName()))
                applicationFields.add(f);

        for (Method delegateMethod:delegateMethods)
        {
            try {
                QtActivity.class.getDeclaredMethod(delegateMethod.getName(), delegateMethod.getParameterTypes());
                if (QtApplication.m_delegateMethods.containsKey(delegateMethod.getName()))
                    QtApplication.m_delegateMethods.get(delegateMethod.getName()).add(delegateMethod);
                else
                {
                    ArrayList<Method> delegateSet = new ArrayList<Method>();
                    delegateSet.add(delegateMethod);
                    QtApplication.m_delegateMethods.put(delegateMethod.getName(), delegateSet);
                }
                for(Field applicationField:applicationFields)
                {
                    if (applicationField.getName().equals(delegateMethod.getName()))
                    {
                        try {
```

```
                                applicationField.set(null, delegateMethod);
                            } catch (Exception e) {
                                e.printStackTrace();
                            }
                        }
                    }
                }
                catch (Exception e)
                {
                }
            }
        }

        @Override
        public void onTerminate() {
            if (m_delegateObject != null && m_delegateMethods.containsKey("onTerminate"))
                invokeDelegateMethod(m_delegateMethods.get("onTerminate").get(0));
            super.onTerminate();
        }

        public static class InvokeResult
        {
            public boolean invoked = false;
            public Object methodReturns = null;
        }

        private static int stackDeep=-1;
        public static InvokeResult invokeDelegate(Object... args)
        {
            InvokeResult result = new InvokeResult();
            if (m_delegateObject==null)
                return result;
            StackTraceElement[] elements=Thread.currentThread().getStackTrace();
            if (-1 == stackDeep)
            {
                String activityClassName=QtActivity.class.getCanonicalName();
                for(int it=0;it<elements.length;it++)
                    if (elements[it].getClassName().equals(activityClassName))
                    {
                        stackDeep=it;
                        break;
                    }
            }
            final String methodName=elements[stackDeep].getMethodName();
            if (-1 == stackDeep || !m_delegateMethods.containsKey(methodName))
                return result;

            for (Method m:m_delegateMethods.get(methodName))
                if (m.getParameterTypes().length == args.length)
                {
                    result.methodReturns=invokeDelegateMethod(m, args);
                    result.invoked=true;
                    return result;
                }
            return result;
        }

        public static Object invokeDelegateMethod(Method m, Object... args)
        {
            try {
                return m.invoke(m_delegateObject, args);
            } catch (Exception e) {
                e.printStackTrace();
            }
            return null;
        }
}
```

## 8.33 C++/customIrcClient/chatbox.cpp File Reference

```
#include "chatbox.h"
#include "ui_chatbox.h"
#include <QScrollArea>
```

Include dependency graph for chatbox.cpp:




## Namespaces

- namespace client

## 8.34 chatbox.cpp

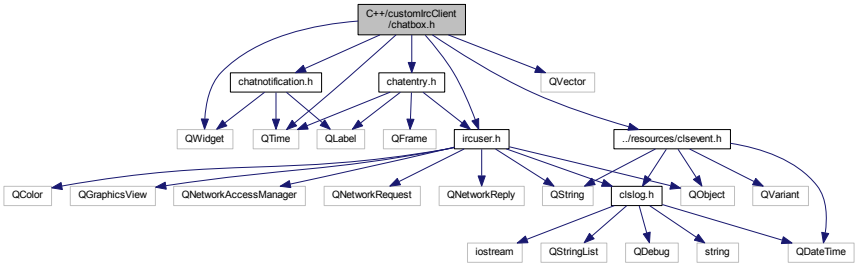
```
#include "chatbox.h"
#include "ui_chatbox.h"
#include <QScrollArea>

namespace client{

    chatBox::chatBox(QWidget *parent) :
        QWidget(parent),
        ui(new Ui::chatBox),
        widthOfNick(0)
    {
        ui->setupUi(this);
    }



    chatBox::~chatBox()
    {
        delete ui;
    }

    void chatBox::addChat(ircUser *user, QString message){
        content *myContent = new content;
        myContent->widget = new chatEntry(user,message,this->widthOfNick,this);
        myContent->type = WIDGETTYPES_CHATENTRY;

        this->entries.append(myContent);
        this->ui->verticalLayout->addWidget(myContent->widget);
    }

    void chatBox::addChat(ircUser *user, QString message, QTime timeOfMessage){
        content *myContent = new content;
        myContent->widget = new chatEntry(user,message, this->widthOfNick, timeOfMessage, this);
        myContent->type = WIDGETTYPES_CHATENTRY;

        this->entries.append(myContent);
        this->ui->verticalLayout->addWidget(myContent->widget);
    }

    void chatBox::addMessage(QString &message){
        this->addMessage(message,QTime::currentTime());
    }

    void chatBox::addMessage(const QString &message){
        QString newMsg(message);
        this->addMessage(newMsg);
    }

```

```
    void chatBox::addMessage(const QString &message, QTime timeOfMessage){
        QString newMsg(message);
        this->addMessage(newMsg, timeOfMessage);
    }

    void chatBox::addMessage(QString &message, QTime timeOfMessage){
        QColor myColor("black");
        this->addMessage(message,timeOfMessage, myColor);
    }

    void chatBox::addMessage(QString &message, QTime timeOfMessage, QColor &myColor){
        content *myContent = new content;
        myContent->widget = new chatNotification(message, timeOfMessage, myColor, this);
        myContent->type = WIDGETTYPES_NOTIFICATION;

        this->entries.append(myContent);
        this->ui->verticalLayout->addWidget(myContent->widget);
    }


    void chatBox::setAllUserNameSize(int newSize){
        this->widthOfNick = newSize;
        foreach (content *ce, this->entries) {
            if(ce->type == WIDGETTYPES_CHATENTRY){
                chatEntry *myWidget = dynamic_cast<chatEntry*>(ce->widget);
                myWidget->setUserWidth(newSize);
            }
        }
    }
}
```

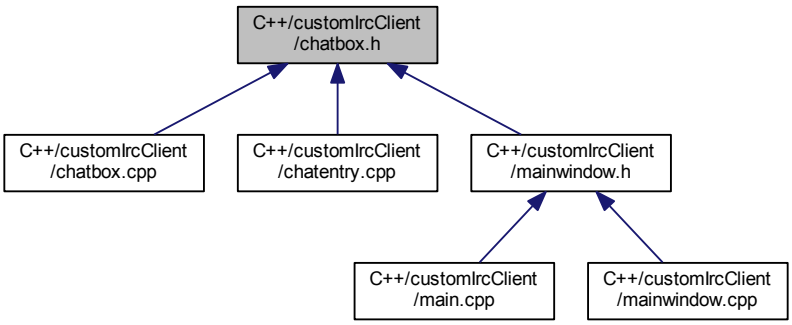
## 8.35 C++/customIrcClient/chatbox.h File Reference

```
#include <QWidget>
#include <QVector>
#include <QTime>
#include "../resources/clsevent.h"
#include "ircuser.h"
#include "chatentry.h"
#include "chatnotification.h"
```

Include dependency graph for chatbox.h:



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




### Classes

- class client::chatBox

  *UI Segment containing the chatbox.*
- struct client::chatBox::content

### Namespaces

- namespace Ui
- namespace client

## 8.36 chatbox.h

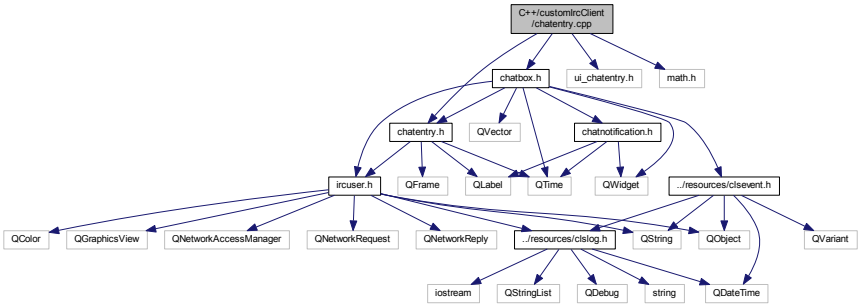
```
#ifndef CHATBOX_H
#define CHATBOX_H

#include <QWidget>
#include <QVector>
#include <QTime>
#include "../resources/clsevent.h"
#include "ircuser.h"

#include "chatentry.h"
#include "chatnotification.h"

namespace Ui {
    class chatBox;
}

namespace client {

    class chatBox : public QWidget
    {
        Q_OBJECT
        enum widgetTypes{
            WIDGETTYPES_CHATENTRY = 0x00,
            WIDGETTYPES_NOTIFICATION
        };
        struct content{
            widgetTypes type;
            QWidget *widget;
        };

    public:
        explicit chatBox(QWidget *parent = 0);
        ~chatBox();

        void addChat(ircUser *user, QString message);
```

```
        void addChat(ircUser *user, QString message, QTime timeOfMessage);

        void addMessage(QString &message);
        void addMessage(const QString &message);

        void addMessage(const QString &message, QTime timeOfMessage);

        void addMessage(QString &message, QTime timeOfMessage);


        void addMessage(QString &message, QTime timeOfMessage, QColor &myColor);


        void setAllUserNameSize(int newSize);

    private:
        Ui::chatBox *ui;
        QVector<content*> entries;
        int widthOfNick;


    };

}
#endif // CHATBOX_H
```

## 8.37 C++/customIrcClient/chatentry.cpp File Reference

```
#include "chatentry.h"
#include "ui_chatentry.h"
#include <math.h>
#include "chatbox.h"
```

Include dependency graph for chatentry.cpp:




### Namespaces

- namespace client

## 8.38 chatentry.cpp

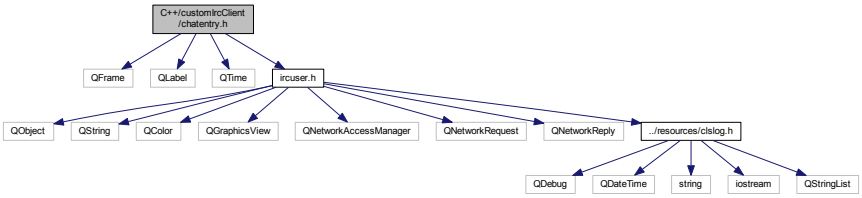
```
#include "chatentry.h"
#include "ui_chatentry.h"
#include <math.h>
#include "chatbox.h"


namespace client{

    chatEntry::chatEntry(ircUser *user, QString message, int width, QWidget *
    parent) :
        QFrame(parent),
        ui(new Ui::chatEntry),
        name(user),
```

```
        message(message),
        timeOfMessage(QTime::currentTime())
    {
        this->init(parent, width);
    }

    void chatEntry::init(QWidget *parent, int width){
        ui->setupUi(this);
        this->setMinimumWidth(parent->width());
        this->ui->lblTime->setText(this->timeOfMessage.toString());
        this->ui->lblNick->setText(this->name->getNick() + ":");
        QPalette palette = ui->lblNick->palette();
        palette.setColor(ui->lblNick->foregroundRole(), this->name->getColor());
        ui->lblNick->setPalette(palette);

        this->message.replace(QRegExp("((?:https?|ftp)://\\S+)"), "<a href=\"\\1\">\\1</a>");

        this->ui->lblChat->setText(this->message);

        this->ui->lblNick->setMinimumWidth(width);
        if(this->getWidth(this->ui->lblNick) > this->ui->lblNick->width()){
            chatBox *pr = qobject_cast<chatBox*>(parent);
            this->setUserWidth(this->getWidth(this->ui->lblNick)+8);
            pr->setAllUserNameSize(this->getWidth(this->ui->lblNick)+8);
        }
        this->calculateMessageWidth();
    }

    chatEntry::chatEntry(ircUser *user, QString message, int width, QTime timeOfMessage, QWidget *parent)
        :QFrame(parent)
         ,ui(new Ui::chatEntry)
         ,name(user)
         ,message(message)
         ,timeOfMessage(timeOfMessage)
    {
        this->init(parent,width);
    }

    void chatEntry::calculateMessageWidth(){
        this->ui->lblChat->setGeometry(this->ui->lblNick->width() +
                this->ui->lblTime->width() + 8,0,
                this->width() -
                (this->ui->lblNick->width() + 16 +
                 this->ui->lblTime->width() + 16),
                16);

        int lines = ceil((double)this->getWidth(this->ui->lblChat) /
                (double)this->ui->lblChat->width());

        this->setMinimumHeight(lines * this->getHeight(this->ui->lblChat));
        this->ui->lblChat->setMinimumHeight(this->minimumHeight() + 2);
    }

    int chatEntry::getWidth(QLabel *l) const{
        return l->fontMetrics().width(l->text());
    }

    int chatEntry::getHeight(QLabel *l) const{
        return l->fontMetrics().height();
    }

    chatEntry::~chatEntry(){
        delete ui;
    }

    void chatEntry::setUserWidth(int newWidth){
        this->ui->lblNick->setMinimumWidth(newWidth);
        this->ui->lblNick->setMaximumWidth(newWidth+2);

        this->calculateMessageWidth();
    }

    void chatEntry::resizeEvent(QResizeEvent *event){
        Q_UNUSED(event);
        this->calculateMessageWidth();
    }
}
```

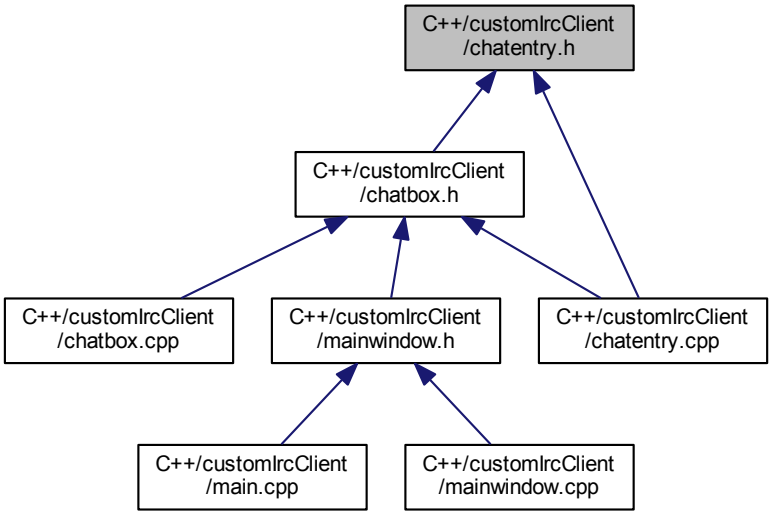
## 8.39 C++/customIrcClient/chatentry.h File Reference

```
#include <QFrame>
#include <QLabel>
#include <QTime>
#include "ircuser.h"
```

Include dependency graph for chatentry.h:




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




### Classes

- class client::chatEntry

  *Class that holds a chatmessage for UI.*

### Namespaces

- namespace Ui
- namespace client

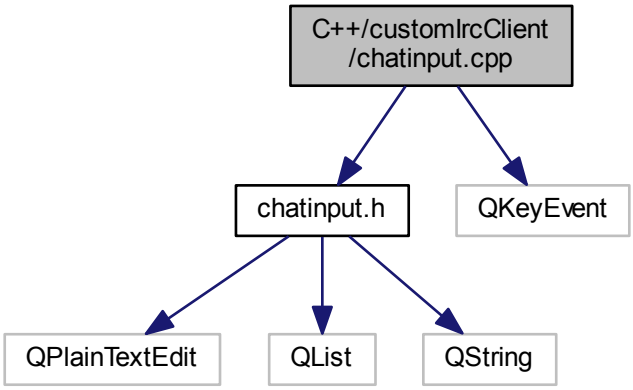
## 8.40 chatentry.h

```
#ifndef CHATENTRY_H
#define CHATENTRY_H

#include <QFrame>
#include <QLabel>
#include <QTime>
#include "ircuser.h"

namespace Ui {
    class chatEntry;
}
namespace client{

    class chatEntry : public QFrame
    {
        Q_OBJECT

        public:
            explicit chatEntry(ircUser *user, QString message, int width,
                    QWidget *parent = 0);
            explicit chatEntry(ircUser *user, QString message, int width,
                    QTime timeOfMessage, QWidget *parent = 0);
            ~chatEntry();

            void setUserWidth(int newWidth);

            void resizeEvent(QResizeEvent *event);


        private:
            Ui::chatEntry *ui;
            ircUser *name;
            QString message;
            int getWidth(QLabel *l) const;
            int getHeight(QLabel *l) const;
            void init(QWidget *parent, int width);
            void calculateMessageWidth();

            QTime timeOfMessage;
    };
}
#endif // CHATENTRY_H
```

## 8.41 C++/customIrcClient/chatinput.cpp File Reference

```
#include "chatinput.h"
#include <QKeyEvent>
```

Include dependency graph for chatinput.cpp:



### Namespaces

- namespace client

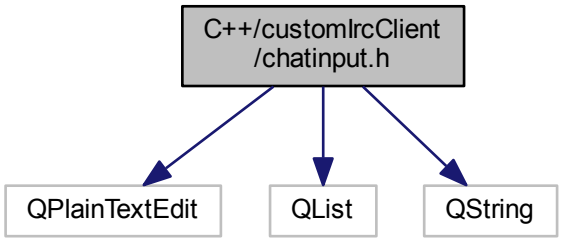
## 8.42 chatinput.cpp

```
#include "chatinput.h"
#include <QKeyEvent>


namespace client{

    chatInput::chatInput(QWidget *parent) :
        QPlainTextEdit(parent)
    {
    }


    void chatInput::keyPressEvent(QKeyEvent *e){
        QTextCursor c = textCursor();
        switch (e->key()) {
            case Qt::Key_Escape:
                // whatever
                break;

            case Qt::Key_Enter:
            case Qt::Key_Return:
                this->checkForCommand(this->toPlainText());
                break;
            default:
                QPlainTextEdit::keyPressEvent(e);
                break;
        }

    }

    void chatInput::checkForCommand(QString text){
        if(text.at(0) == '/'){
            //Yay, a command!
            //Split it
            QList<QString> splits = text.split(' ');
            splits[0] = splits[0].mid(1);//To remove the /
            QString command = splits[0];
            command = command.toLower();
            splits.pop_front(); //Remove command
            emit this->commandGiven(command,splits);
        }else{
            emit this->sendChat(text);
        }
        this->setPlainText("");
    }

}
```

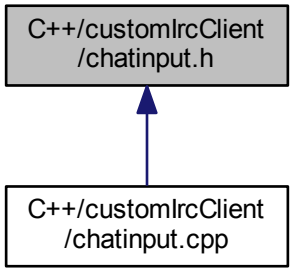
## 8.43 C++/customIrcClient/chatinput.h File Reference

```
#include <QPlainTextEdit>
#include <QList>
#include <QString>
```

Include dependency graph for chatinput.h:




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




### Classes

- class client::chatInput

  *UI segment containing the chat input.*

### Namespaces

- namespace client

## 8.44 chatinput.h

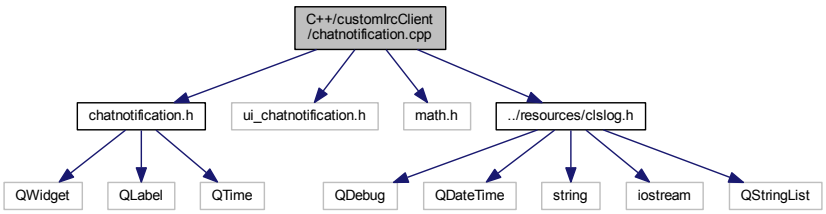
```
00001 #ifndef CHATINPUT_H
00002 #define CHATINPUT_H
00003
00004 #include <QPlainTextEdit>
00005 #include <QList>
00006 #include <QString>
00007
00008 namespace client{
00013     class chatInput : public QPlainTextEdit
00014     {
```

```
        Q_OBJECT
        public:
            explicit chatInput(QWidget *parent = 0);


        Q_SIGNALS:
            void sendChat(QString message);
            void commandGiven(QString command, QList<QString> args);

        public slots:


        private:
            void keyPressEvent(QKeyEvent *e);
            void checkForCommand(QString text);
    };
}
#endif // CHATINPUT_H
```

## 8.45 C++/customIrcClient/chatnotification.cpp File Reference

```
#include "chatnotification.h"
#include "ui_chatnotification.h"
#include <math.h>
#include "../resources/clslog.h"
```

Include dependency graph for chatnotification.cpp:




### Namespaces

- namespace client

## 8.46 chatnotification.cpp

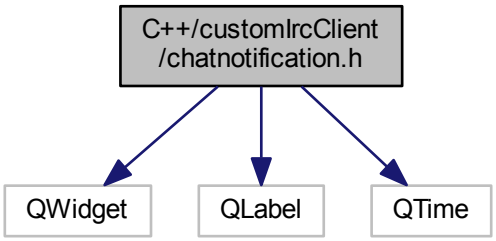
```
#include "chatnotification.h"
#include "ui_chatnotification.h"
#include <math.h>
#include "../resources/clslog.h"


namespace client{
    chatNotification::chatNotification(QString message, QWidget *parent)
        :QWidget(parent)
         ,ui(new Ui::chatNotification)
         ,message(message)
         ,timeOfMessage(QTime::currentTime())
    {
        ui->setupUi(this);
        this->setMinimumWidth(parent->width());

        this->ui->lblTime->setText(this->timeOfMessage.toString());

        this->ui->lblNotification->setText(this->message);

        this->calculateMessageWidth();
    }
```

```
00023
00024     chatNotification::chatNotification(QString message, QTime
      timeOfMessage, QColor myColor, QWidget *parent)
00025         :QWidget(parent)
00026         ,ui(new Ui::chatNotification)
00027         ,message(message)
00028         ,timeOfMessage(timeOfMessage)
00029     {
00030         ui->setupUi(this);
00031         this->setMinimumWidth(parent->width());
00032
00033         this->ui->lblTime->setText(this->timeOfMessage.toString());
00034
00035         QPalette palette = ui->lblNotification->palette();
00036         palette.setColor(ui->lblNotification->foregroundRole(), myColor);
00037         ui->lblNotification->setPalette(palette);
00038
00039         this->ui->lblNotification->setText(this->message);
00040
00041         this->calculateMessageWidth();
00042     }
00043
00044     chatNotification::~chatNotification()
00045     {
00046         delete ui;
00047     }
00048
00049     void chatNotification::calculateMessageWidth(){
00050         this->ui->lblNotification->setGeometry(this->ui->lblTime->width() + 4,0,
00051                 this->width() -
00052                 (this->ui->lblTime->width() + 16),
00053                 16);
00054
00055         int lines = ceil((double)this->getWidth(this->ui->lblNotification) /
00056                 (double)this->ui->lblNotification->width());
00057
00058         this->setMinimumHeight(lines * this->getHeight(this->ui->lblNotification));
00059         this->ui->lblNotification->setMinimumHeight(this->minimumHeight() + 4);
00060     }
00061
00062     int chatNotification::getWidth(QLabel *l) const{
00063         return l->fontMetrics().width(l->text());
00064     }
00065
00066     int chatNotification::getHeight(QLabel *l) const{
00067         return l->fontMetrics().height();
00068     }
00069
00070
00071
00072     void chatNotification::resizeEvent(QResizeEvent *event){
00073         Q_UNUSED(event);
00074         this->calculateMessageWidth();
00075     }
00076
00077 }
```

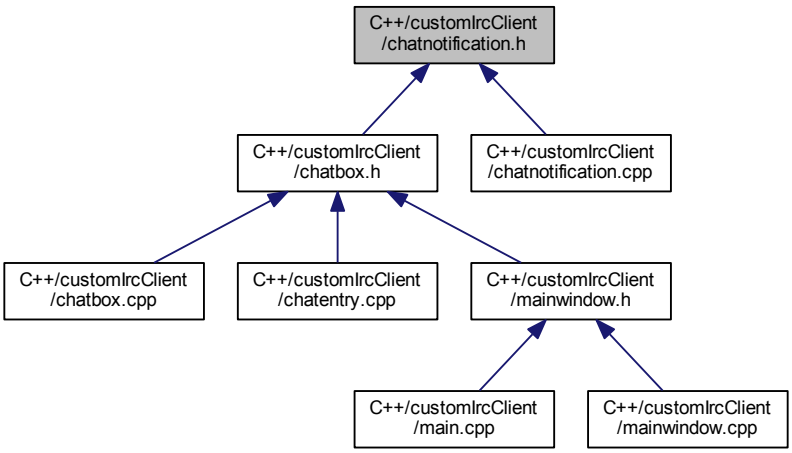
## 8.47 C++/customIrcClient/chatnotification.h File Reference

```
#include <QWidget>
#include <QLabel>
#include <QTime>
```

Include dependency graph for chatnotification.h:




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




### Classes

- class client::chatNotification

  *UI Segment containing a notification in the chatbox.*

### Namespaces

- namespace Ui
- namespace client

## 8.48 chatnotification.h

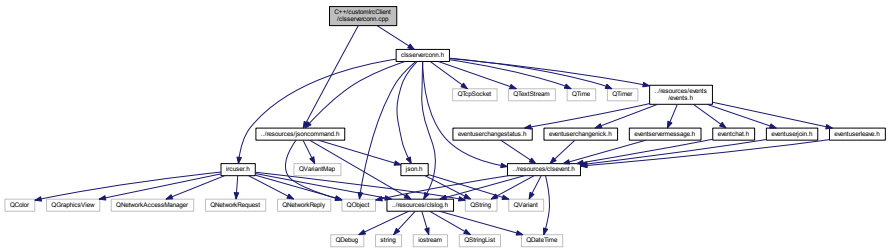
```
00001 #ifndef CHATNOTIFICATION_H
00002 #define CHATNOTIFICATION_H
```

```
00003
00004 #include <QWidget>
00005 #include <QLabel>
00006 #include <QTime>
00007
00008 namespace Ui {
00009     class chatNotification;
00010 }
00011 namespace client{
00012
00017     class chatNotification : public QWidget
00018     {
00019         Q_OBJECT
00020
00021         public:
00027             explicit chatNotification(QString message, QWidget *parent = 0);
00028
00036             explicit chatNotification(QString message, QTime
      timeOfMessage,
00037                                       QColor myColor, QWidget *parent = 0);
00038             ~chatNotification();
00039
00040             void resizeEvent(QResizeEvent *event);
00041         private:
00045             void calculateMessageWidth();
00046             int getHeight(QLabel *l) const;
00047             int getWidth(QLabel *l) const;
00048
00049             Ui::chatNotification *ui;
00050             QString message;
00051             QTime timeOfMessage;
00052     };
00053 }
00054 #endif // CHATNOTIFICATION_H
```

## 8.49 C++/customIrcClient/clsserverconn.cpp File Reference

```
#include "clsserverconn.h"
#include "../resources/jsoncommand.h"
```

Include dependency graph for clsserverconn.cpp:




### Namespaces

- namespace client

## 8.50 clsserverconn.cpp

```
00001 #include "clsserverconn.h"
00002 #include "../resources/jsoncommand.h"
00003
00004 namespace client{
00005
00006     clsServerConn::clsServerConn(const QString hostName, const int port,
00007             QObject *parent)
00008         :QObject(parent)
00009         ,hostName(hostName)
00010         ,port(port)
00011         ,depth(0)
```

```
        ,log(LOGTAGS_UI)
        ,isConnected(false)
        ,isConnecting(false)
    {
        this->buffer = new QString();

        this->sock = new QTcpSocket(this);

        this->sock->connect(this->sock, SIGNAL(readyRead()), this,
                SLOT(readData()));
        this->sock->connect(this->sock, SIGNAL(connected()), this,
                SLOT(srvConnected()));
        this->sock->connect(this->sock, SIGNAL(disconnected()), this,
                SLOT(srvDisconnected()));
    }

    void clsServerConn::doConnect(){
        this->connectTimer->singleShot(MAX_CONNECT_TIMEOUT_IN_MS, this
, SLOT(connectTimeout()));
        this->isConnecting = true;
        this->isConnected = false;
        this->sock->connectToHost(this->hostName,this->port);


        this->log << "Connecting" << endl;
    }

    void clsServerConn::sendCommand(jsonCommand &toSend){
        //this->log << "Sending json object: " << docToSend->toJson() << endl;
        this->log << "Sending JSON: " << toSend << endl;
        QByteArray bAToSend;
        bAToSend.append(toSend.toJsonString() + "\r\n");
        this->sock->write(bAToSend);
    }

    void clsServerConn::srvConnected(){
        this->log << "Connected!" << endl;
        this->isConnecting = false;
        this->isConnected = true;
        emit this->connected();
    }


    void clsServerConn::srvDisconnected(){
        this->log << "Disconnected" << endl;
        this->isConnecting = false;
        this->isConnected = false;
        emit this->disconnected();
    }

    void clsServerConn::handleCompletedQuery(
jsonCommand &comm){
        QVector<ircUser*> users;
        foreach(QVariant user, comm.getData("users").toList()){
            users.append(new ircUser(user));
        }
        emit this->userQueryCompleted(users);
    }

    void clsServerConn::handleOwnUserChange(
jsonCommand &comm){
        //This function might have the same body as handleUserInfo.
        //But for now this is the only thing changeable.
        //Maybe later there will be more stuff added here
        if(comm.getData("change") == "STATUS"){
            //Own user is always ID 1..
            //FIXME: Own user no longer receives updates. Probaby move this to an event
            //emit this->userStatusChange(1, comm.getData("new").toString());
        }
    }

    void clsServerConn::handleEventRequest(
jsonCommand &comm){
        //Let them flow, all the chats!
        QList<QVariant> events = comm.getData("events").toList();
        foreach(QVariant myEvent, events){
            this->handleEventReceived(myEvent.toMap());
        }
    }

    void clsServerConn::handleEventReceived(QVariantMap data){
        clsEvent myEvent(data);
        switch(myEvent.getType()){
            case EVENTTYPE_CHAT:
                emit this->chatReceived(new eventChat(data));
                break;
            case EVENTTYPE_USER_JOIN:
```

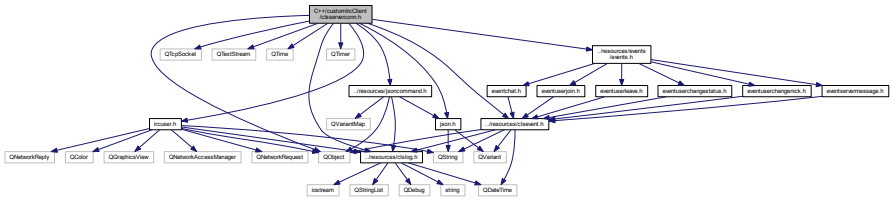
```
                emit this->userEnter(new eventUserJoin(data));
                break;
            case EVENTTYPE_USER_LEFT:
                emit this->userLeave(new eventUserLeave(data));
                break;
            case EVENTTYPE_USER_STATUS:
                emit this->userStatusChange(new eventUserChangeStatus(data));
                break;
            case EVENTTYPE_USER_NICK:
                emit this->userChangeNick(new eventUserChangeNick(data));
                break;
            case EVENTTYPE_SERVERMSG:
                emit this->serverMessageReceived(new eventServerMessage(data));
                break;
            default:
                break; //ERROR!
        }

    }


    bool clsServerConn::checkForJsonCommand(QByteArray toAdd, jsonCommand &comm){
        bool ret = false;
        foreach(char b, toAdd){
            if(b == '{' || b == '['){
                depth++;
            }else if(b == '}' || b == ']'){
                depth--;
            }
            this->buffer->append(b);

            if(depth == 0){
                //Clear the buffer.
                if(comm.getCommand() == JSONCOMMAND_NONE){
                    ret = true;
                    comm = new jsonCommand(*this->buffer);
                    this->log << "Data received: \t" << this->buffer << endl;
                    this->buffer->clear();
                }
            }
        }
        return ret;
    }

    void clsServerConn::readData(){
        jsonCommand comm(JSONCOMMAND_NONE);
        if(this->checkForJsonCommand(this->sock->readAll(), comm)){
            switch (comm.getCommand()) {
                case JSONCOMMAND_EVENT:
                    this->handleEventReceived(comm.getData("event").toMap());
                    break;
                case JSONCOMMAND_USERQUERY:
                    this->handleCompletedQuery(comm);
                    break;
                case JSONCOMMAND_CHANGEOWNUSER:
                    this->handleOwnUserChange(comm);
                    break;
                case JSONCOMMAND_GETEVENTS:
                    this->handleEventRequest(comm);
                default:
                    break;
            }
        }
    }

    void clsServerConn::connectTimeout(){
        if(this->isConnecting && this->isConnected == false){
            this->sock->abort();
            emit this->sgnConnectTimeout();
        }

    }

}
```

## 8.51 C++/customIrcClient/clsserverconn.h File Reference

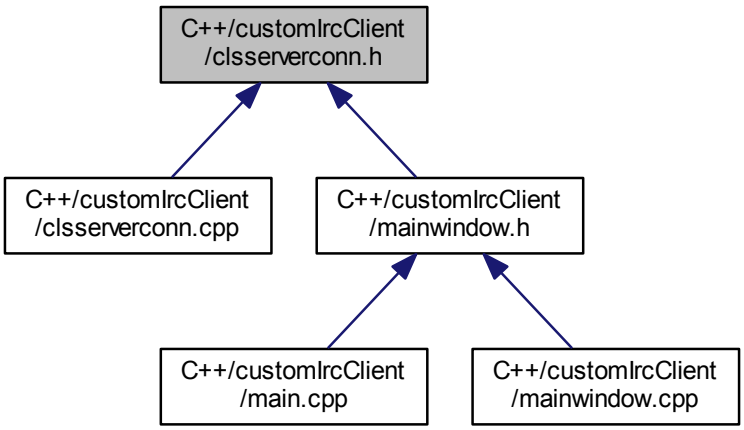
```
#include <QObject>
```

```
#include <QTcpSocket>
#include <QTextStream>
#include <QTime>
#include "../resources/clslog.h"
#include <QTimer>
#include "ircuser.h"
#include "../resources/jsoncommand.h"
#include "../resources/json.h"
#include "../resources/clsevent.h"
#include "../resources/events/events.h"
```

Include dependency graph for clsserverconn.h:




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




## Classes

- class client::clsServerConn

  *Class that has a connection to the server.*

## Namespaces

- namespace client

### Macros

- #define MAX_CONNECT_TIMEOUT_IN_MS 5000

### 8.51.1 Macro Definition Documentation

#### 8.51.1.1 #define MAX_CONNECT_TIMEOUT_IN_MS 5000

Definition at line 17 of file clsserverconn.h.

## 8.52 clsserverconn.h

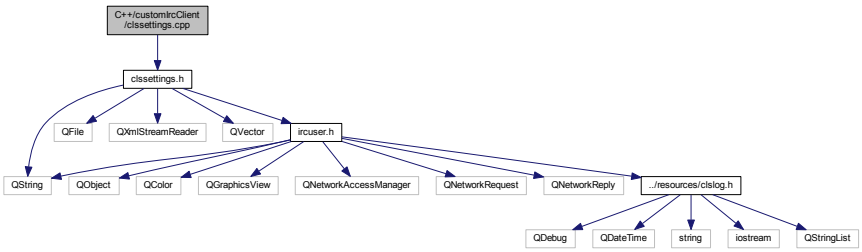
```
00001 #ifndef CLSSERVERCONN_H
00002 #define CLSSERVERCONN_H
00003
00004 #include <QObject>
00005 #include <QTcpSocket>
00006 #include <QTextStream>
00007 #include <QTime>
00008 #include "../resources/clslog.h"
00009 #include <QTimer>
00010 #include "ircuser.h"
00011 #include "../resources/jsoncommand.h"
00012 #include "../resources/json.h"
00013 #include "../resources/clsevent.h"
00014 #include "../resources/events/events.h"
00015
00016
00017 #define MAX_CONNECT_TIMEOUT_IN_MS 5000
00018
00019 using namespace shared;
00020 using namespace shared::events;
00021 namespace client{
00022
00027     class clsServerConn : public QObject
00028     {
00029         Q_OBJECT
00030         public:
00038             explicit clsServerConn(const QString hostName, const int port,
00039                     QObject *parent = 0);
00040
00044             void doConnect();
00049             void sendCommand(jsonCommand &toSend);
00050
00051
00052
00053         private:
00065             bool checkForJsonCommand(QByteArray toAdd, jsonCommand &comm);
00070             void handleCompletedQuery(jsonCommand &comm);
00071
00076             void handleOwnUserChange(jsonCommand &comm);
00077
00082             void handleEventRequest(jsonCommand &comm);
00083
00088             void handleEventReceived(QVariantMap data);
00089
00090 signals:
00095             void chatReceived(eventChat* event);
00099             void connected();
00103             void disconnected();
00104
00109             void userQueryCompleted(QVector<ircUser*> users);
00110
00116             void userStatusChange(eventUserChangeStatus *changeEvent);
00117
00123             void userChangeNick(eventUserChangeNick *changeEvent);
00124
00129             void userLeave(eventUserLeave *leaveEvent);
00130
00135             void userEnter(eventUserJoin *newUser);
00136
00141             void serverMessageReceived(eventServerMessage *theMessage);
00142
00146             void sgnConnectTimeout();
00147         public slots:
00148             void srvDisconnected();
00149             void srvConnected();
00150             void readData();
```

```
            void connectTimeout();

        private:
            QString hostName;
            int port;

            QTcpSocket *sock;
            QString *buffer;
            int depth;

            clsLog log;

            QTimer *connectTimer;
            bool isConnected;
            bool isConnecting;


    };
}
#endif // CLSSERVERCONN_H
```

## 8.53 C++/customIrcClient/clssettings.cpp File Reference

`#include "clssettings.h"`

Include dependency graph for clssettings.cpp:




### Namespaces

- namespace client

## 8.54 clssettings.cpp

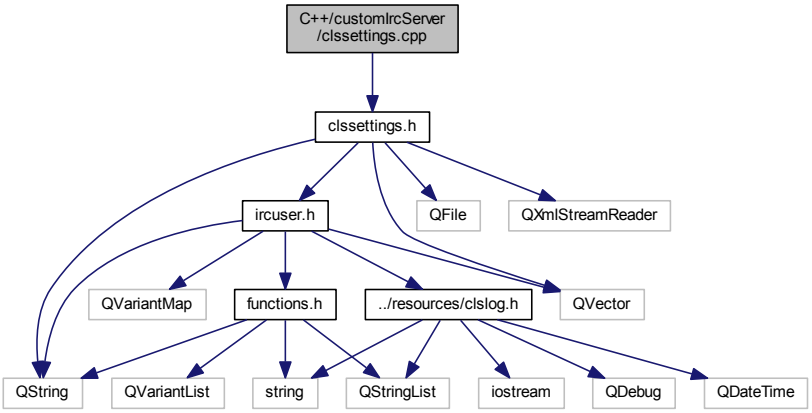
```
#include "clssettings.h"

namespace client{

    clsSettings::clsSettings(const QString &filePath){
        QString fileName = filePath + "/settings.xml";
        QXmlStreamReader Rxml;
        QFile file(fileName);
        if (!file.open(QFile::ReadOnly | QFile::Text)){
            std::cerr << "Error: Cannot read file " << fileName.toStdString()
                << ": " << qPrintable(file.errorString())
                << std::endl;
        }
        Rxml.setDevice(&file);
        Rxml.readNext();


        while(!Rxml.atEnd()){
            if(Rxml.isStartElement()){
                if(Rxml.name() == "settings"){
                    Rxml.readNext(); //Start
                } else if(Rxml.name() == "server"){
                    Rxml.readNext();
                    this->readServer(Rxml);
                }
```

```
            }
            Rxml.readNext();
        }
        file.close();
    }

    QString clsSettings::getServer() const{
        return this->server;
    }

    int clsSettings::getPort() const{
        return this->port;
    }


    void clsSettings::readServer(QXmlStreamReader &reader){
        while(!reader.isEndElement()){
            QString name = reader.name().toString();
            if(name == "server"){
                this->server = reader.readElementText();
            }else if(name == "port"){
                this->port = reader.readElementText().toInt();
            }
            reader.readNext();
        }
    }

}
```

## 8.55 C++/customIrcServer/clssettings.cpp File Reference

`#include "clssettings.h"`

Include dependency graph for clssettings.cpp:




### Namespaces

- namespace server

## 8.56 clssettings.cpp

```
#include "clssettings.h"

namespace server {
clsSettings::clsSettings(const QString &filePath)
    :listeningPort(DEFAULT_LISTEN_PORT)
{
    QString fileName = filePath + "/settings.xml";
```

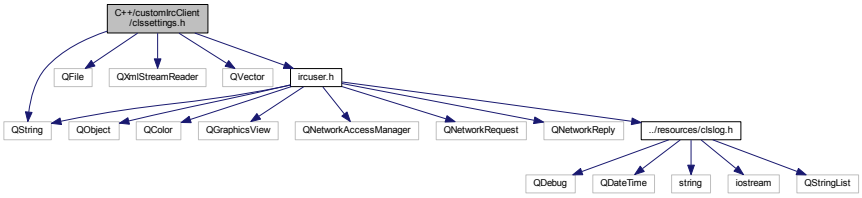
```
    QXmlStreamReader Rxml;
    QFile file(fileName);
    if (!file.open(QFile::ReadOnly | QFile::Text)){
        std::cerr << "Error: Cannot read file " << fileName.toStdString()
         << ": " << qPrintable(file.errorString())
         << std::endl;
    }
    Rxml.setDevice(&file);
    Rxml.readNext();


    while(!Rxml.atEnd()){
        if(Rxml.isStartElement()){
            if(Rxml.name() == "settings"){
                Rxml.readNext(); //Start
            } else if(Rxml.name() == "server"){
                Rxml.readNext();
                this->readServer(Rxml);
            } else if(Rxml.name() == "users"){
                Rxml.readNext();
                this->readUsers(Rxml);
            } else if(Rxml.name() == "userInfo"){
                Rxml.readNext();
                this->readUserInfo(Rxml);
            } else if(Rxml.name() == "uiServer"){
                Rxml.readNext();
                this->readUiSettings(Rxml);
            }
        }
        Rxml.readNext();
   }
    file.close();
}

int clsSettings::getListeningPort() const{
    return this->listeningPort;
}

QString clsSettings::getServer() const{
    return this->server;
}

QString clsSettings::getChannel() const{
    return this->channel;
}

QString clsSettings::getUserName() const{
    return this->ownUser->getName();
}

ircUser *clsSettings::getOwnUser() const{
    return this->ownUser;
}

QVector<ircUser *> clsSettings::getUsers() const{
    return this->users;
}

void clsSettings::readUiSettings(QXmlStreamReader &reader){
    while(!reader.isEndElement()){
        QString name = reader.name().toString();
        if(name == "listenPort"){
            this->listeningPort = reader.readElementText().toInt();
        }
        reader.readNext();
    }
}


void clsSettings::readServer(QXmlStreamReader &reader){
    while(!reader.isEndElement()){
        QString name = reader.name().toString();
        if(name == "server"){
            this->server = reader.readElementText();
        }else if(name == "channel"){
            this->channel = reader.readElementText();
        }
        reader.readNext();
    }
}

void clsSettings::readUsers(QXmlStreamReader &reader){
    while(!reader.isEndElement()){
        if(reader.name().toString() == "user"){
            reader.readNext();
            QString username;
            QString email;
```

```
            bool endElement = reader.isEndElement();
            while(!endElement){
                endElement = reader.isEndElement();
                QString name = reader.name().toString();
                if(name == "username"){
                    username = reader.readElementText();
                }else if(name == "email"){
                    email = reader.readElementText();
                }
                reader.readNext();
            }
            ircUser *myUser = new ircUser(username,true);
            myUser->setEmail(email);

            this->users.append(myUser);
        }
        reader.readNext();
    }
}

void clsSettings::readUserInfo(QXmlStreamReader &reader){
    QString username;
    QString email;
    while(!reader.isEndElement()){
        QString name = reader.name().toString();
        if(name == "username"){
            username = reader.readElementText();
        }else if(name == "email"){
            email = reader.readElementText();
        }
        reader.readNext();
    }
    this->ownUser = new ircUser(username,true);
    this->ownUser->setEmail(email);
    this->ownUser->setOnline(true);
}


} /*End of namespace*/
```

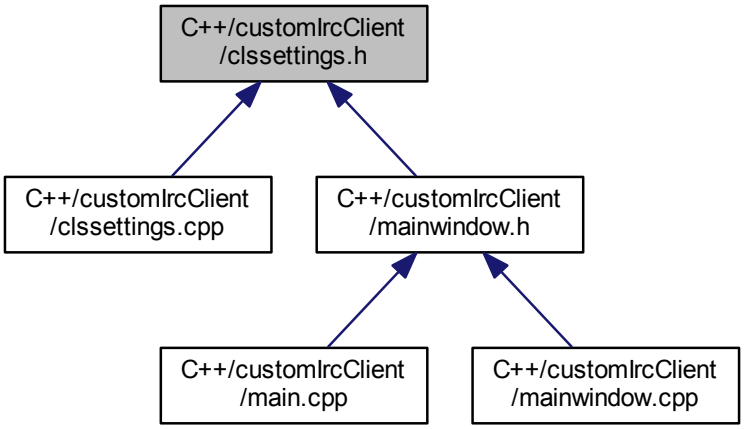
## 8.57 C++/customIrcClient/clssettings.h File Reference

```
#include <QString>
#include <QFile>
#include <QXmlStreamReader>
#include <QVector>
#include "ircuser.h"
```

Include dependency graph for clssettings.h:



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




### Classes

- class client::clsSettings

  *Settings class containing all the variables needed for the client.*

### Namespaces

- namespace client

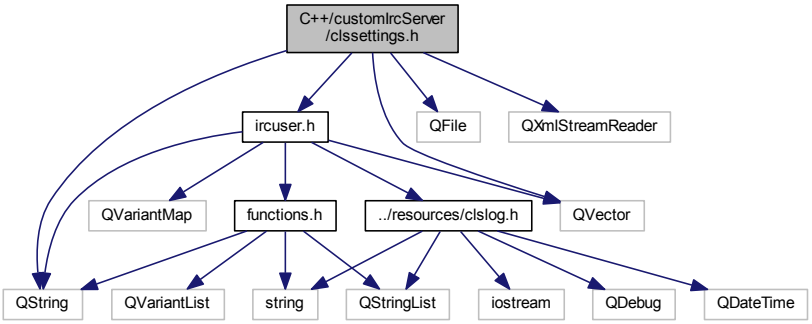
## 8.58 clssettings.h

```
#ifndef CLSSETTINGS_H
#define CLSSETTINGS_H

#include <QString>
#include <QFile>
#include <QXmlStreamReader>
#include <QVector>
#include "ircuser.h"

namespace client{
    class clsSettings
    {
        public:
            clsSettings(const QString &filePath);

            QString getServer() const;

            int getPort() const;
        private: //Functions
            void readServer(QXmlStreamReader &reader);


        private: //Values
            QString server;
            int port;
    };
}
#endif // CLSSETTINGS_H
```

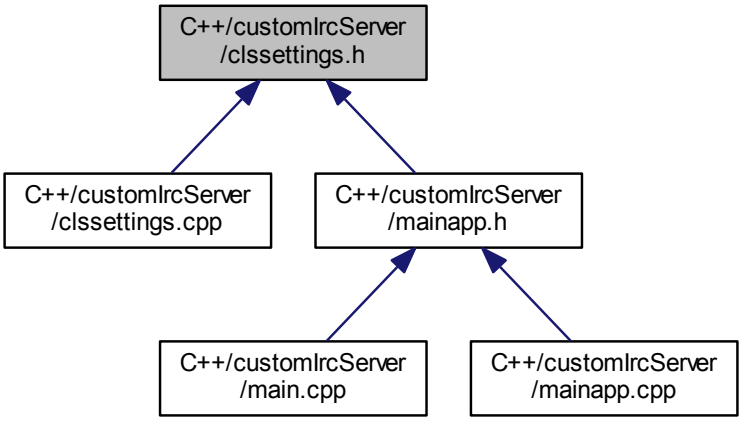
## 8.59 C++/customIrcServer/clssettings.h File Reference

```
#include <QString>
#include <QFile>
#include <QXmlStreamReader>
#include <QVector>
#include "ircuser.h"
```

Include dependency graph for clssettings.h:




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




### Classes

- class server::clsSettings

  *Settings class.*

### Namespaces

- namespace server

### Macros

- #define DEFAULT_LISTEN_PORT 1337

### 8.59.1 Macro Definition Documentation

#### 8.59.1.1 #define DEFAULT_LISTEN_PORT 1337

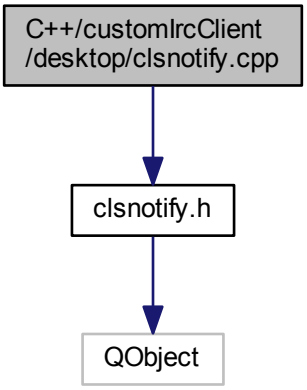
Definition at line 10 of file clssettings.h.

## 8.60 clssettings.h

```
00001 #ifndef CLSSETTINGS_H
00002 #define CLSSETTINGS_H
00003
00004 #include <QString>
00005 #include <QFile>
00006 #include <QXmlStreamReader>
00007 #include <QVector>
00008 #include "ircuser.h"
00009
00010 #define DEFAULT_LISTEN_PORT 1337
00011
00012 namespace server {
00019 class clsSettings
00020 {
00021 public:
00026     clsSettings(const QString &filePath);
00027
00032     int getListeningPort() const;
00033
00038     QString getServer() const;
00039
00044     QString getChannel() const;
00045
00050     QString getUserName() const;
00051
00056     ircUser *getOwnUser() const;
00057
00062     QVector<ircUser *> getUsers() const;
00063 private: //Functions
00069     void readUiSettings(QXmlStreamReader &reader);
00070
00076     void readServer(QXmlStreamReader &reader);
00077
00083     void readUsers(QXmlStreamReader &reader);
00084
00090     void readUserInfo(QXmlStreamReader &reader);
00091
00092
00093 private: //Values
00098     QString server;
00099
00104     QString channel;
00105
00110     ircUser *ownUser;
00111
00116     int listeningPort;
00117
00122     QVector<ircUser*> users;
00123
00124 };
00125 }
00126 #endif // CLSSETTINGS_H
```

## 8.61 C++/customIrcClient/desktop/clsnotify.cpp File Reference

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




### Namespaces

- namespace client

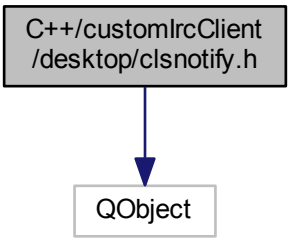
## 8.62 clsnotify.cpp

```
00001 #include "clsnotify.h"
00002
00003 namespace client{
00004
00005     clsNotify::clsNotify(const QString title, const QString body, QObject *parent) :
00006         QObject(parent)
00007     {
00008 #ifdef Q_OS_LINUX
00009         notify_init ("customIrcClient");
00010         this->thisNot = notify_notification_new (title.toStdString().c_str(),
00011                 body.toStdString().c_str(),
00012                 "dialog-information");
00013 #endif
00014     }
00015
00016     void clsNotify::show() const{
00017 #ifdef Q_OS_LINUX
00018         notify_notification_show (this->thisNot, NULL);
00019 #endif
00020     }
00021
00022 }
```

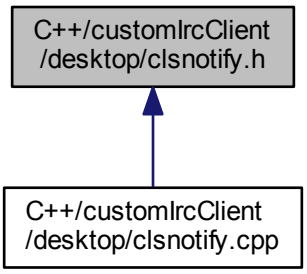
## 8.63 C++/customIrcClient/desktop/clsnotify.h File Reference

```
#include <QObject>
```

Include dependency graph for clsnotify.h:




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




### Classes

- class client::clsNotify

  *Notification class.*

### Namespaces

- namespace client

## 8.64 clsnotify.h

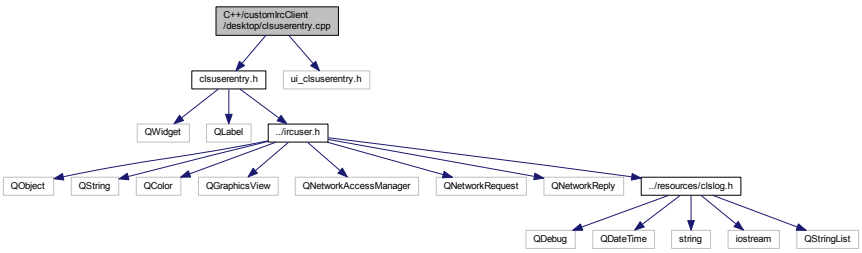
```
00001 #ifndef CLSNOTIFY_H
00002 #define CLSNOTIFY_H
00003
00004 //#define QT_NO_KEYWORDS// To prevent prevent symbol collisions between GTK+ and Qt
```

```
#include <QObject>
#ifdef Q_OS_LINUX
#undef signals
#include <libnotify/notify.h>
#else

#endif
namespace client{
    class clsNotify : public QObject
    {
        Q_OBJECT
        public:
            explicit clsNotify(const QString title, const QString body, QObject *parent = 0);

            void show() const;
            public slots:


        private:
#ifdef Q_OS_LINUX
            NotifyNotification *thisNot;
#endif
    };

}

#endif // CLSNOTIFY_H
```

## 8.65 C++/customIrcClient/desktop/clsuserentry.cpp File Reference

```
#include "clsuserentry.h"
#include "ui_clsuserentry.h"
```

Include dependency graph for clsuserentry.cpp:




### Namespaces

- namespace client

## 8.66 clsuserentry.cpp

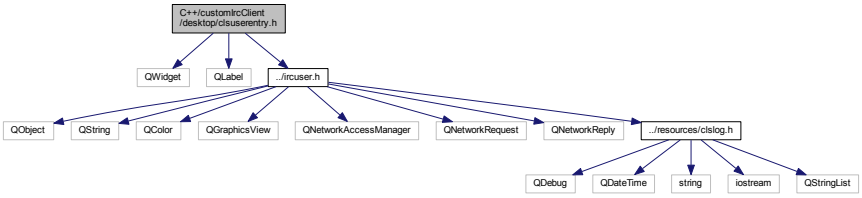
```
#include "clsuserentry.h"
#include "ui_clsuserentry.h"

namespace client{

    clsUserEntry::clsUserEntry(ircUser *user, QWidget *parent)
        :QWidget(parent)
         ,ui(new Ui::clsUserEntry)
         ,user(user)
    {
        ui->setupUi(this);
        this->ui->lblUserName->setText(this->user->getNick());
        QString status = this->user->getStatus();
        if(status == ""){
            status = "Online";
        }
```

```
        this->ui->lblUserStatus->setText(status);

        QPalette palette = ui->lblUserName->palette();
        palette.setColor(ui->lblUserName->foregroundRole(), this->user->getColor());
        ui->lblUserName->setPalette(palette);

        connect(this->user, SIGNAL(iconChanged()), this, SLOT(iconChanged()));
        this->ui->lblGravatar->setPixmap(this->user->getIcon().scaled(
                    this->ui->lblGravatar->size()));
    }

    clsUserEntry::~clsUserEntry(){
        delete ui;
    }

    void clsUserEntry::refresh(){
        this->ui->lblUserName->setText(this->user->getNick());
        QString status = this->user->getStatus();
        if(status == ""){
            status = "Online";
        }

        this->ui->lblUserStatus->setText(status);

        QPalette palette = ui->lblUserName->palette();
        palette.setColor(ui->lblUserName->foregroundRole(), this->user->getColor());
        ui->lblUserName->setPalette(palette);

        this->ui->lblGravatar->setPixmap(this->user->getIcon().scaled(
                    this->ui->lblGravatar->size()));
    }

    void clsUserEntry::iconChanged(){
        this->ui->lblGravatar->setPixmap(this->user->getIcon().scaled(
                    this->ui->lblGravatar->size()));
    }

}
```

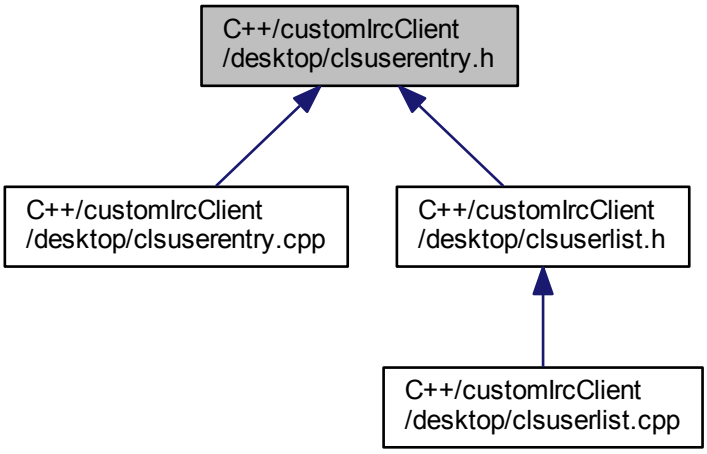
## 8.67 C++/customIrcClient/desktop/clsuserentry.h File Reference

```
#include <QWidget>
#include <QLabel>
#include "../ircuser.h"
```

Include dependency graph for clsuserentry.h:



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




### Classes

- class client::clsUserEntry

  *UI segment for a user entry in the user list.*

### Namespaces

- namespace Ui
- namespace client

## 8.68 clsuserentry.h

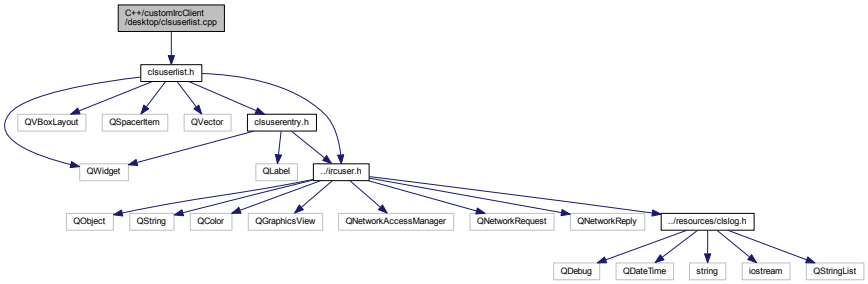
```
#ifndef CLSUSERENTRY_H
#define CLSUSERENTRY_H

#include <QWidget>
#include <QLabel>
#include "../ircuser.h"

namespace Ui {
    class clsUserEntry;
}

namespace client{
    class clsUserEntry : public QWidget
    {
        Q_OBJECT

        public:
            explicit clsUserEntry(ircUser *user, QWidget *parent = 0);
            ~clsUserEntry();

            void refresh();
        private slots:
            void iconChanged();

        private:
            Ui::clsUserEntry *ui;

```

```
            ircUser *user;
    };

}
#endif // CLSUSERENTRY_H
```

## 8.69 C++/customIrcClient/desktop/clsuserlist.cpp File Reference

`#include "clsuserlist.h"`

Include dependency graph for clsuserlist.cpp:




### Namespaces

- namespace client

## 8.70 clsuserlist.cpp

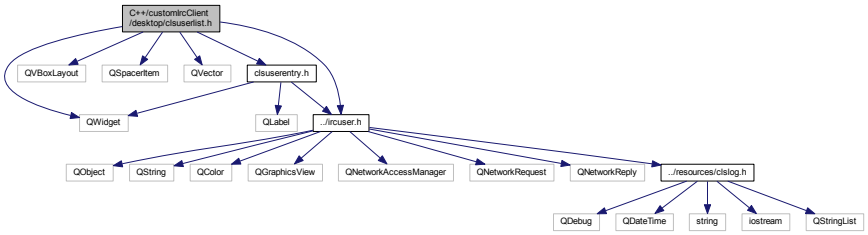
```
#include "clsuserlist.h"

namespace client{

    clsUserList::clsUserList(QWidget *parent)
        :QWidget(parent)
    {
        this->setupUi();
    }

    clsUserList::~clsUserList()
    {
    }

    void clsUserList::addUser(ircUser *user){
        //TODO: Alphabetical order
        this->verticalLayout->removeItem(this->verticalSpacer);

        clsUserEntry *nE = new clsUserEntry(user);

        this->verticalLayout->addWidget(nE);
        this->users.insert(user,nE);

        this->verticalLayout->addItem(this->verticalSpacer);
        this->setMinimumHeight(this->users.count() * nE->minimumHeight());
    }

    void clsUserList::refreshUser(ircUser *user){
        this->users[user]->refresh();
    }

    void clsUserList::removeUser(ircUser *user){
        clsUserEntry *ourUser = this->users[user];
        delete this->users[user];
        //this->verticalLayout->removeWidget(ourUser);
        //this->users.remove(user);
    }

```

```
    void clsUserList::clear(){
        this->setupUi();
    }

    void clsUserList::setupUi(){
        if (this->objectName().isEmpty())
            this->setObjectName(QString("clsUserList"));
        this->resize(400, 300);
        QSizePolicy sizePolicy(QSizePolicy::Ignored, QSizePolicy::Ignored);
        sizePolicy.setHorizontalStretch(0);
        sizePolicy.setVerticalStretch(0);
        sizePolicy.setHeightForWidth(this->sizePolicy().hasHeightForWidth());
        this->setSizePolicy(sizePolicy);
        verticalLayout = new QVBoxLayout(this);
        verticalLayout->setSpacing(0);
        verticalLayout->setObjectName(QString("verticalLayout"));
        verticalLayout->setContentsMargins(0, 0, 0, 0);
        verticalSpacer = new QSpacerItem(20, 40, QSizePolicy::Minimum, QSizePolicy::Expanding
    );

        verticalLayout->addItem(verticalSpacer);
    }
}
```

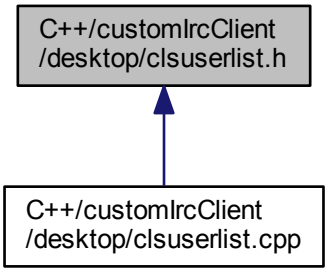
## 8.71 C++/customIrcClient/desktop/clsuserlist.h File Reference

```
#include <QWidget>
#include <QVBoxLayout>
#include <QSpacerItem>
#include <QVector>
#include "clsuserentry.h"
#include "../ircuser.h"
```

Include dependency graph for clsuserlist.h:




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



### Classes

- class client::clsUserList

  *UI Segment for user list.*

### Namespaces

- namespace Ui
- namespace client

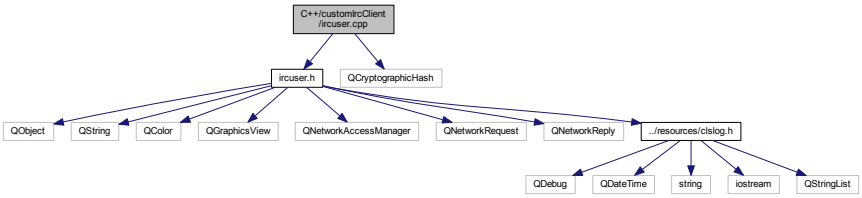
## 8.72 clsuserlist.h

```
00001 #ifndef clsUserList_H
00002 #define clsUserList_H
00003
00004 #include <QWidget>
00005 #include <QVBoxLayout>
00006 #include <QSpacerItem>
00007 #include <QWidget>
00008 #include <QVector>
00009 #include "clsuserentry.h"
00010 #include "../ircuser.h"
00011
00012 namespace Ui {
00013     class clsUserList;
00014 }
00015 namespace client{
00016
00021     class clsUserList : public QWidget
00022     {
00023         Q_OBJECT
00024
00025         public:
00026             explicit clsUserList(QWidget *parent = 0);
00027             ~clsUserList();
00028
00033             void addUser(ircUser *user);
00038             void refreshUser(ircUser *user);
00043             void removeUser(ircUser *user);
00044
00048             void clear();
00049         private:
00050             void setupUi();
00051             QVBoxLayout *verticalLayout;
00052             QSpacerItem *verticalSpacer;
00053
00054             QMap<ircUser*,clsUserEntry*> users;
00055
00056
00057     };
00058 }
00059 #endif // clsUserList_H
```

## 8.73 C++/customIrcClient/ircuser.cpp File Reference

```
#include "ircuser.h"
#include <QCryptographicHash>
```

Include dependency graph for ircuser.cpp:




## Namespaces

- namespace client

## 8.74 ircuser.cpp

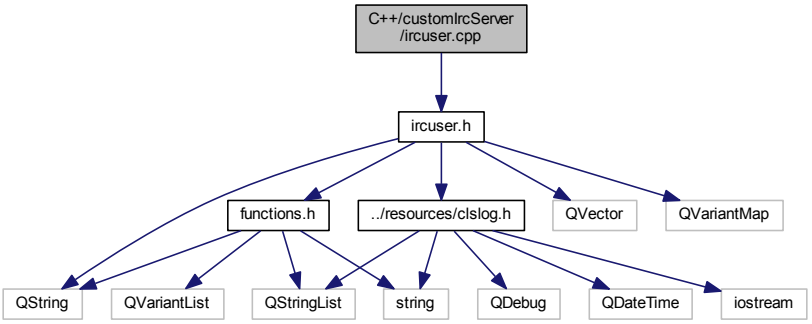
```
#include "ircuser.h"
#include <QCryptographicHash>

namespace client{

    ircUser::~ircUser(){
    }


    ircUser::ircUser(QVariant jsonData)
        :log(LOGTAGS_USER)
    {
        QVariantMap data = jsonData.toMap();
        this->id = data["id"].toInt();
        this->nick = data["nick"].toString();
        this->status = data["status"].toString();
        this->standard = data["standard"].toBool();
        this->uColor = QColor(data["uColor"].toMap()["r"].toInt(),
                data["uColor"].toMap()["g"].toInt(),
                data["uColor"].toMap()["b"].toInt());
        this->email = data["email"].toString();
        this->online = data["online"].toBool();
        //Fetch gravatar here plox
        //Fetch gravatar
        QNetworkAccessManager *m_netwManager = new QNetworkAccessManager();
        connect(m_netwManager, SIGNAL(finished(QNetworkReply*)), this,
                SLOT(slot_netwManagerFinished(QNetworkReply*)));

        QString hash = QString(
                QCryptographicHash::hash(this->email.trimmed().toLower().toUtf8(),
                    QCryptographicHash::Md5).toHex());
        QUrl url("http://www.gravatar.com/avatar/" + hash);
        QNetworkRequest request(url);
        m_netwManager->get(request);
    }

    ircUser::ircUser(qint32 id, QString nick, QString status, QColor uColor, bool standard)
        :id(id)
         ,nick(nick)
         ,status(status)
         ,uColor(uColor)
         ,standard(standard)
         ,log(LOGTAGS_USER)
    {}

    void ircUser::copyFrom(ircUser *userToCopy){
        this->nick = userToCopy->nick;
        this->status = userToCopy->status;
        this->uColor = userToCopy->uColor;
        this->online = userToCopy->online;
    }

    const QString ircUser::getNick() const{
        return this->nick;
    }

```

```
    qint32 ircUser::getId() const{
        return this->id;
    }

    const QString ircUser::getStatus() const{
        return this->status;
    }

    const QColor ircUser::getColor() const{
        return this->uColor;
    }

    const QPixmap ircUser::getIcon() const{
        return this->icon;
    }

    bool ircUser::getStandard() const{
        return this->standard;
    }

    bool ircUser::getOnline() const{
        return this->online;
    }

    void ircUser::setNick(const QString &newNick){
        this->nick = newNick;
    }

    void ircUser::setStatus(const QString &newStatus){
        if(newStatus == ""){
            this->status = "Online";
        }else{
            this->status = newStatus;
        }
    }

    void ircUser::setColor(const QColor &newColor){
        this->uColor = newColor;
    }


    void ircUser::slot_netwManagerFinished(QNetworkReply *reply){
        if (reply->error() != QNetworkReply::NoError) {
            qDebug() << "Error in" << reply->url() << ":" << reply->errorString();
            return;
        }

        QByteArray jpegData = reply->readAll();
        this->icon.loadFromData(jpegData);
        //this->ui->lblGravatar->setPixmap(pixmap.scaled(this->ui->lblGravatar->size()));
        emit this->iconChanged();
    }


    void ircUser::setOnline(const bool isOnline){
        this->online = isOnline;
        if(this->online == false){
            this->status = "Offline";
        }
    }

}
```

## 8.75 C++/customIrcServer/ircuser.cpp File Reference

```
#include "ircuser.h"
```

Include dependency graph for ircuser.cpp:




### Namespaces

- namespace server

### Functions

- clsLog & server::operator<< (clsLog &log, const ircUser *user)
- ircUser::userColor server::getColor (int length)

## 8.76 ircuser.cpp

```
00001 #include "ircuser.h"
00002
00003 namespace server {
00004
00005 ircUser::ircUser(const nickAndStatus name, const QString ircId)
00006     :name(name.nick)
00007     ,ircId(ircId)
00008     ,id(0)
00009     ,status(name.status)
00010     ,online(true)
00011     ,standard(false)
00012     ,log(LOGTAGS_USER)
00013 {
00014     if(this->name.left(1) == ":"){
00015         this->name = this->name.mid(1);
00016     }
00017     this->online = false;
00018     if(this->name.indexOf("|") >= 0){
00019         this->status = this->name.mid(this->name.indexOf("|")+1);
00020         this->name = this->name.left(this->name.indexOf("|"));
00021     }
00022     this->uColor.r = 0;
00023     this->uColor.g = 0;
00024     this->uColor.b = 0;
00025 }
00026
00027 ircUser::ircUser(const nickAndStatus name)
00028     :name(name.nick)
00029     ,ircId("NaN")
00030     ,id(0)
00031     ,status(name.status)
00032     ,online(true)
00033     ,standard(false)
00034     ,log(LOGTAGS_USER)
00035 {
00036     if(this->name.left(1) == ":"){
00037         this->name = this->name.mid(1);
00038     }
00039     this->online = false;
```

```
    if(this->name.indexOf("|") >= 0){
        this->status = this->name.mid(this->name.indexOf("|")+1);
        this->name = this->name.left(this->name.indexOf("|"));
    }
    this->uColor.r = 0;
    this->uColor.g = 0;
    this->uColor.b = 0;
}

ircUser::ircUser(const QString name, const bool standard)
    :name(name)
    ,ircId("NaN")
    ,id(0)
    ,status("Offline")
    ,online(false)
    ,standard(standard)
    ,log(LOGTAGS_USER)
{
    if(this->name.left(1) == ":"){
        this->name = this->name.mid(1);
    }
    if(!standard){
        this->online = true;
        this->status = "";
    }

    if(this->name.indexOf("|") >= 0){
        this->status = this->name.mid(this->name.indexOf("|")+1);
        this->name = this->name.left(this->name.indexOf("|"));
    }

    this->uColor.r = 0;
    this->uColor.g = 0;
    this->uColor.b = 0;
}


ircUser::ircUser()
    :name("NaN")
    ,ircId("NaN")
    ,id(0)
    ,status("Offline")
    ,online(false)
    ,standard(false)
    ,log(LOGTAGS_USER)
{
    this->uColor.r = 0;
    this->uColor.g = 0;
    this->uColor.b = 0;
}

QString ircUser::getName() const{
    return this->name;
}

QString ircUser::getStatus() const{
    return this->status;
}

bool ircUser::isStandard() const{
    return this->standard;
}

QVariantMap ircUser::toVariantMap() const{
    QVariantMap ret;
    QVariantMap color;
    color["r"] = this->uColor.r;
    color["g"] = this->uColor.g;
    color["b"] = this->uColor.b;

    ret["uColor"] = QVariant(color);
    ret["nick"] = this->name;
    ret["ircId"] = this->ircId;
    ret["id"] = this->id;
    ret["status"] = this->status;
    ret["standard"] = this->standard;
    ret["isOnline"] = this->online;
    ret["email"] = this->email;

    return ret;

}

void ircUser::setOnline(const bool isOnline){
    this->online = isOnline;
}

```

```
00127 bool ircUser::getOnline() const{
00128     return this->online;
00129 }
00130
00131 void ircUser::setStatus(const QString &newStatus){
00132     this->status = newStatus;
00133 }
00134
00135 void ircUser::setIrcId(const QString &newId){
00136     this->ircId = newId;
00137 }
00138
00139 void ircUser::setNick(const QString &newNick){
00140     this->name = newNick;
00141 }
00142
00143 void ircUser::setColor(const int r, const int g, const int b){
00144     this->uColor.r = r;
00145     this->uColor.g = g;
00146     this->uColor.b = b;
00147 }
00148
00149 void ircUser::setColor(const ircUser::userColor &newColor){
00150     this->uColor = newColor;
00151 }
00152
00153 void ircUser::setEmail(const QString &email){
00154     this->email = email;
00155 }
00156
00157
00158 QString ircUser::getIrcId() const{
00159     return this->ircId;
00160 }
00161
00162 int ircUser::getId() const {
00163     return this->id;
00164 }
00165
00166 void ircUser::setId(const int &newId){
00167     this->id = newId;
00168 }
00169
00170 clsLog &operator <<(clsLog &log, const ircUser *user){
00171     log << "USER: name: \t" << user->name << "\tStatus:\t" << user->status;
00172     log << "\tStandard: \t" << user->standard << endl;
00173     return log;
00174 }
00175
00176
00177 ircUserList::ircUserList(){
00178     this->lastId = 0;
00179 }
00180
00181 ircUser::userColor getColor(int length){
00182     ircUser::userColor retColor;
00183     switch(length){
00184         case 0:
00185             retColor.r = 160;
00186             retColor.g = 25;
00187             retColor.b = 25;
00188             break;
00189         case 1:
00190             retColor.r = 25;
00191             retColor.g = 25;
00192             retColor.b = 160;
00193             break;
00194         case 2:
00195             retColor.r = 25;
00196             retColor.g = 160;
00197             retColor.b = 25;
00198             break;
00199         case 3:
00200             retColor.r = 160;
00201             retColor.g = 160;
00202             retColor.b = 25;
00203             break;
00204         case 4:
00205             retColor.r = 25;
00206             retColor.g = 160;
00207             retColor.b = 160;
00208             break;
00209         case 5:
00210             retColor.r = 255;
00211             retColor.g = 0;
00212             retColor.b = 0;
00213             break;
```

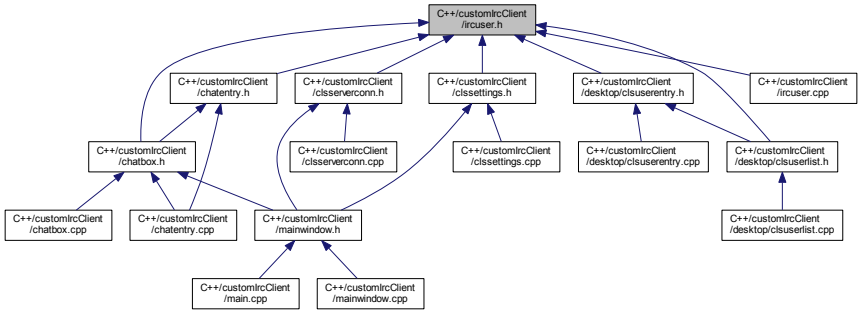
```
        case 6:
            retColor.r = 0;
            retColor.g = 160;
            retColor.b = 0;
            break;
        case 7:
            retColor.r = 0;
            retColor.g = 0;
            retColor.b = 255;
            break;
        default:
            retColor.r = 0;
            retColor.g = 0;
            retColor.b = 0;
            break;
    }
    return retColor;
}

void ircUserList::add(ircUser *userToAdd){
    userToAdd->setColor(getColor(this->lastId % 8));
    this->lastId++;
    userToAdd->setId(this->lastId);
    this->users.append(userToAdd);
}

void ircUserList::del(ircUser *userToDel){
    for(int i = 0; i < this->users.size(); i++){
        if(this->users[i]->getName() == userToDel->getName()){
            this->users.remove(i);
        }
    }
}

ircUser *ircUserList::getUser(const nickAndStatus nick, const QString ircId){
    ircUser *searched = this->getUserByIrcId(ircId);
    if(searched->getName() == "NaN"){
        searched = this->getUserByNick(nick);
        if(searched->getName() == "NaN"){
            return new ircUser;
        }else{
            return searched;
        }
    }else{
        return searched;
    }
}

ircUser *ircUserList::getUserByNick(nickAndStatus nick) {
    for(int i=0; i < this->users.size(); i++){
        if(this->users[i]->getName() == nick.nick){
            return &*this->users[i];
        }
    }
    return new ircUser();
}

ircUser *ircUserList::getUserByIrcId(QString ircId){
    for(int i=0; i < this->users.size(); i++){
        if(this->users[i]->getIrcId() == ircId){
            return &*this->users[i];
        }
    }
    return new ircUser();
}

QVector<ircUser *> ircUserList::getAll() const{
    return this->users;
}

ircUser *ircUser::operator =(const ircUser *user){
    this->id = user->id;
    this->email = user->email;
    this->ircId = user->ircId;
    this->name = user->name;
    this->online = user->online;
    this->standard = user->standard;
    this->status = user->status;
    this->uColor = user->uColor;

    return this;
}


} /*End of namespace*/
```

## 8.77 C++/customIrcClient/ircuser.h File Reference

```
#include <QObject>
#include <QString>
#include <QColor>
#include <QGraphicsView>
#include <QNetworkAccessManager>
#include <QNetworkRequest>
#include <QNetworkReply>
#include "../resources/clslog.h"
```

Include dependency graph for ircuser.h:

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




### Classes

- class client::ircUser

  *The ircUser class.*

### Namespaces

- namespace client

## 8.78 ircuser.h

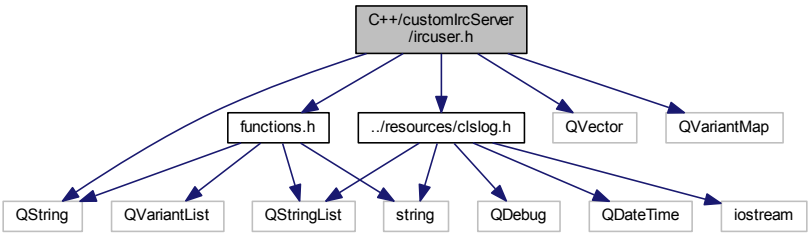
```
00001 #ifndef IRCUSER_H
00002 #define IRCUSER_H
00003
00004 #include <QObject>
00005 #include <QString>
00006 #include <QColor>
00007 #include <QGraphicsView>
00008 #include <QNetworkAccessManager>
00009 #include <QNetworkRequest>
```

```
#include <QNetworkReply>
#include "../resources/clslog.h"

using namespace K4U;
namespace client{

    class ircUser : public QObject
    {
        Q_OBJECT

        public:
            explicit ircUser(QVariant jsonData);
            explicit ircUser(qint32 id, QString nick, QString status, QColor uColor, bool standard);

            void copyFrom(ircUser *userToCopy);
            ~ircUser();

            //Getters
            const QString getNick() const;
            qint32 getId() const;
            const QString getStatus() const;
            const QColor getColor() const;
            const QString getEmail() const;
            const QPixmap getIcon() const;
            bool getStandard() const;
            bool getOnline() const;

            //Setters
            void setNick(const QString &newNick); //Probably never used
            void setStatus(const QString &newStatus);
            void setColor(const QColor &newColor); //Also probably never used
            void setEmail(const QString &email);
            void setOnline(const bool isOnline);



        signals:
            void iconChanged() const;

        private slots:
                void slot_netwManagerFinished(QNetworkReply* reply);

        private:
            qint32 id;
            QString nick;
            QString status;
            QColor uColor;
            bool standard;
            QString email;
            QPixmap icon;
            bool online;

            clsLog log;
    };
}
#endif // IRCUSER_H
```

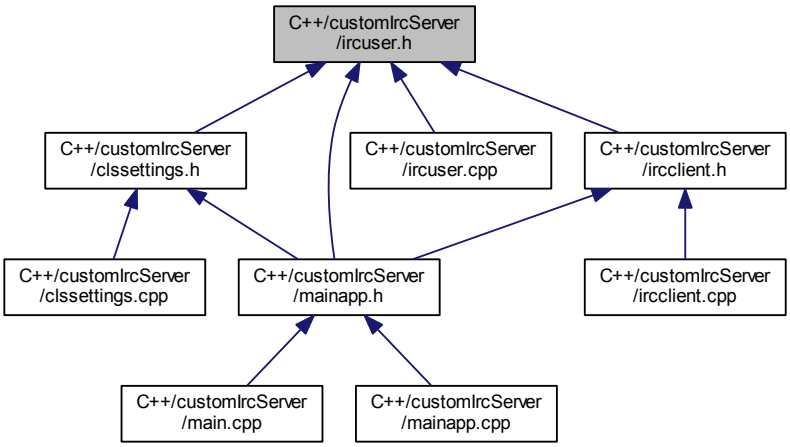
## 8.79 C++/customIrcServer/ircuser.h File Reference

```
#include <QString>
#include <QVector>
#include <QVariantMap>
#include "functions.h"
#include "../resources/clslog.h"
```

Include dependency graph for ircuser.h:




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




### Classes

- class server::ircUser

  *The ircUser class.*
- struct server::ircUser::userColor

  *The userColor struct As a replacement for the QColor type.*
- class server::ircUserList

  *Contains all the users.*

### Namespaces

- namespace server

## 8.80 ircuser.h

```
00001 #ifndef IRCUSER_H
00002 #define IRCUSER_H
```

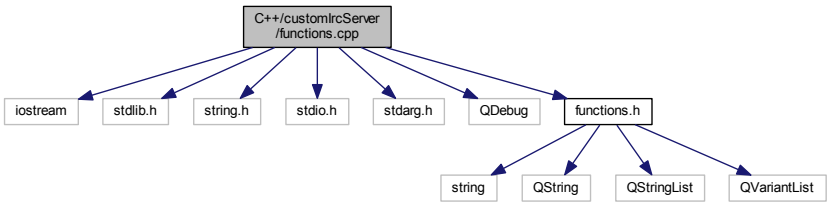
```
00003
00004 #include <QString>
00005 #include <QVector>
00006 #include <QVariantMap>
00007 #include "functions.h"
00008 #include "../resources/clslog.h"
00009
00010 using namespace K4U;
00011 namespace server {
00019 class ircUser {
00020
00021 public:
00026     struct userColor{
00027         int r;
00028         int g;
00029         int b;
00030     };
00031
00037     ircUser(const nickAndStatus name, const QString ircId);
00038
00045     ircUser(const QString name, const bool standard);
00046
00053     ircUser(const nickAndStatus name);
00054
00055     ircUser();
00056
00057     //Getters
00058     QString getName() const;
00059     bool getOnline() const;
00060     QString getStatus() const;
00061     bool isStandard() const;
00066     QVariantMap toVariantMap() const;
00067
00073     QString getIrcId() const;
00079     int getId() const;
00080
00081     //Setters
00082     void setOnline(const bool isOnline);
00083     void setStatus(const QString &newStatus);
00084     void setIrcId(const QString &newId);
00085     void setId(const int &newId);
00086     void setNick(const QString &newNick);
00087     void setColor(const int r, const int g, const int b);
00088     void setColor(const userColor &newColor);
00089     void setEmail(const QString &email);
00090
00091
00092     friend clsLog& operator<<(clsLog& log, const ircUser* user);
00093
00094     ircUser* operator=(const ircUser* user);
00095 private:
00096     QString name;
00097     QString ircId;
00098     int id;
00099     QString status;
00100     QString email;
00101     bool online;
00102     bool standard;
00103
00104     userColor uColor;
00105
00106     clsLog log;
00107 };
00108
00113 class ircUserList {
00114 public:
00115     ircUserList();
00116
00121     void add(ircUser *userToAdd);
00126     void del(ircUser *userToDel);
00127
00133     ircUser *getUserByNick(nickAndStatus nick);
00140     ircUser *getUser(const nickAndStatus nick, const QString ircId);
00141
00148     ircUser *findUser(ircUser &userToFind);
00149
00156     ircUser *getUserById(int id);
00163     ircUser *getUserByIrcId(QString ircId);
00164
00169     QVector<ircUser*> getAll() const;
00170
00171
00172 private:
00173     QVector<ircUser*> users;
00174     int lastId;
00175 };
00176
```

```
00177 }
00178 #endif // IRCUSER_H
```

## 8.81 C++/customIrcServer/functions.cpp File Reference

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <stdarg.h>
#include <QDebug>
#include "functions.h"
```

Include dependency graph for functions.cpp:




### Namespaces

- namespace server

### Functions

- const string server::qts (const QString str)

  *Shorthand to convert a QString to a std::string.*
- nickAndStatus server::getNickAndStatus (const QString dNick)

  *Converts a nickname to a nickAndStatus (nick|status)*
- const QString server::getId (const QString fullId)

  *Converts a full IRC-ID to something without the nickname.*
- bool server::inCommand (const QString *command, const char *search)

  *Just an instring command. Checks if search is in command.*
- nickAndStatus server::getNickAndStatusFromId (const QString id)

  *Converts an IRC-ID to a nickAndStatus (nick|status)*

## 8.82 functions.cpp

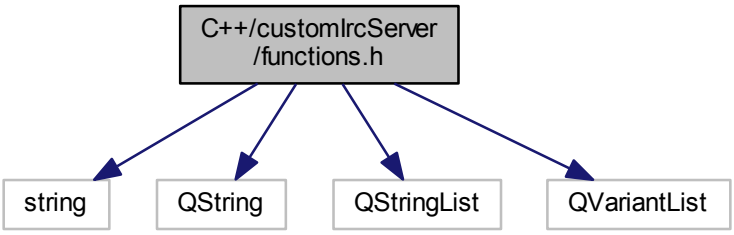
```
00001
00002 #include <iostream>
00003 #include <stdlib.h>
00004 #include <string.h>
00005 #include <stdio.h>
00006 #include <stdarg.h>
00007 #include <QDebug>
00008
00009 #include "functions.h"
00010
00011 using namespace std;
```

```
00012 namespace server{
00013
00014 const string qts(const QString str){
00015     return str.toStdString();
00016 }
00017
00018 nickAndStatus getNickAndStatus(const QString dNick){
00019     nickAndStatus ret;
00020     QString nick = dNick;
00021
00022     if(dNick.indexOf("!") >= 0){
00023         nick = dNick.left(nick.indexOf("!"));
00024     }
00025     if(nick.indexOf("|") >= 0){
00026         ret.nick = nick.left(nick.indexOf("|"));
00027         ret.status =  nick.mid(nick.indexOf("|")+1);
00028     }else{
00029         ret.nick = nick;
00030     }
00031     if(ret.nick.left(1) == ":"){
00032         ret.nick = ret.nick.mid(1);
00033     }
00034     return ret;
00035 }
00036
00037 const QString getId(const QString full){
00038     return full.mid(full.indexOf("!")+1);
00039 }
00040
00041 bool inCommand(const QString *command, const char* search){
00042     return (command->indexOf(search) >= 0);
00043 }
00044
00045
00046 nickAndStatus getNickAndStatusFromId(const QString id){
00047     QString nick(id.left(id.indexOf("!")));
00048     return getNickAndStatus(nick);
00049 }
00050
00051
00052
00053 }
```

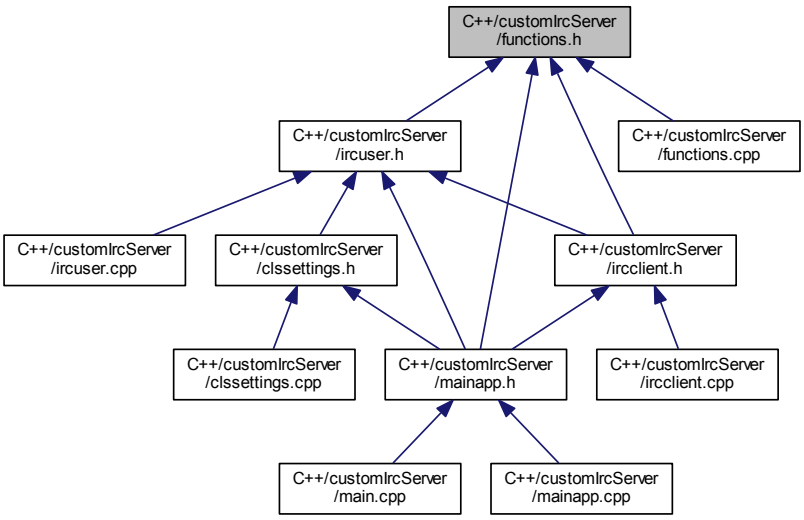
## 8.83 C++/customIrcServer/functions.h File Reference

```
#include <string>
#include <QString>
#include <QStringList>
#include <QVariantList>
```

Include dependency graph for functions.h:



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




### Classes

- struct server::nickAndStatus

  *The nickAndStatus struct.*

### Namespaces

- namespace server

### Functions

- bool server::inCommand (const QString *command, const char *search)

  *Just an instring command. Checks if search is in command.*
- const string server::qts (const QString str)

  *Shorthand to convert a QString to a std::string.*
- nickAndStatus server::getNickAndStatus (const QString dNick)

  *Converts a nickname to a nickAndStatus (nick|status)*
- const QString server::getId (const QString fullId)

  *Converts a full IRC-ID to something without the nickname.*
- nickAndStatus server::getNickAndStatusFromId (const QString id)

  *Converts an IRC-ID to a nickAndStatus (nick|status)*

## 8.84 functions.h

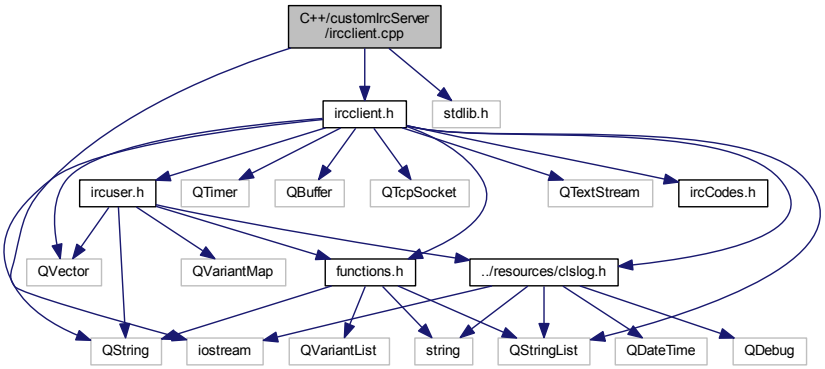
```
00001 #ifndef FUNCTIONS_H
00002 #define FUNCTIONS_H
00003 #include <string>
00004 #include <QString>
00005 #include <QStringList>
00006 #include <QVariantList>
```

```
00007
00008 namespace server{
00013 struct nickAndStatus{
00014     QString nick;
00015     QString status;
00016 };
00017
00024 extern bool inCommand(const QString *command, const char* search);
00025
00031 extern const std::string qts(const QString str);
00032
00038 extern nickAndStatus getNickAndStatus(const QString dNick);
00039
00045 extern const QString getId(const QString fullId);
00046
00052 extern nickAndStatus getNickAndStatusFromId(const QString id);
00053
00054
00055
00056 }
00057 #endif // FUNCTIONS_H
```

## 8.85 C++/customIrcServer/ircclient.cpp File Reference

```
#include <iostream>
#include <stdlib.h>
#include "ircclient.h"
```

Include dependency graph for ircclient.cpp:




### Namespaces

- namespace server

## 8.86 ircclient.cpp

```
00001 #include <iostream>
00002 #include <stdlib.h>
00003 #include "ircclient.h"
00004
00005 using namespace std;
00006 namespace server {
00007
00008 ircClient::ircClient(const QString &hostname, const QString &channel
00009                      , const QString &nick, const int port)
00010     :hostname(QString(hostname))
00011     ,port(port)
00012     ,channel(QString(channel))
00013     ,log(clsLog(LOGTAGS_IRC))
```

```
    ,connected(false)
    ,connecting(false)
    ,suppressDisconnectSignal(false)
{
    this->ownNickAndStatus.nick = nick;
    this->init();
}

void ircClient::init(){
    this->log << "Initializing IRC Client for host " << this->hostname << ":";
    this->log << this->port << endl;

    buffer = new QBuffer(this);
    buffer->open(QIODevice::ReadWrite);

    this->timeOutChecker = new QTimer(this);
    this->connectTimer = new QTimer(this);

    this->sock = new QTcpSocket(this);
    this->sock->connect(this->sock, SIGNAL(readyRead()), this,
                        SLOT(readData()));
    this->sock->connect(this->sock, SIGNAL(connected()), this,
                        SLOT(sltConnected()));
    this->sock->connect(this->sock, SIGNAL(disconnected()), this,
                        SLOT(disconnected()));

    this->timeOutChecker->connect(this->timeOutChecker, SIGNAL(timeout()),
                                  this, SLOT(checkForPing()));
    this->connectTimer->connect(this->connectTimer, SIGNAL(timeout()),
                                this, SLOT(connectTimeout()));
}

void ircClient::disconnected(){
    this->log << "Disconnected" << endl;
    if(suppressDisconnectSignal == false){
        emit this->IRCDisconnected("Socket disconnect");
    }
    suppressDisconnectSignal = false;
}


ircClient::~ircClient(){
    buffer->close();
    this->sock->close();
}

void ircClient::doConnect(){
    this->log << "Connecting" << endl;
    //Set timeout

    this->connected = false;
    this->connecting = true;
    //this->connectTimer->setInterval(MAX_CONNECT_TIMEOUT_IN_MS);
    //this->connectTimer->setSingleShot(true);
    //this->connectTimer->start();
    this->connectTimer->singleShot(MAX_CONNECT_TIMEOUT_IN_MS, this,
SLOT(connectTimeout()));


    this->sock->connectToHost(this->hostname,this->port);
}

void ircClient::setOwnUser(ircUser *ownUser){
    this->ownUser = ownUser;
}

void ircClient::sltConnected(){
    this->connected = true;
    this->connecting = false;
    this->connectTimer->stop();
    this->timeOutChecker->start(PING_CHECK_INTERVAL*1000);
    //doLog(this->TAG, "Connected!");
    this->send(QString("USER %1 0 * :%1").arg(this->ownNickAndStatus.
nick));
    this->send(QString("NICK %1").arg(this->ownNickAndStatus.
nick));

    emit this->IRCConnected();

}



void ircClient::handleCommand(const QString &sender, const QStringList &command){
    if(sender == this->hostname){
        //Message from the server:
        ircCodes::ircCode code = (ircCodes::ircCode) atoi(
```

```
qts(command[0]).c_str());
        QString userNick;
        QString userId;
        switch (code) {
        case ircCodes::RPL_ENDOFMOTD:
        case ircCodes::ERR_NOMOTD:
            //OH YES!
            this->motdReceived = true;
            this->send(QString("JOIN #%1").arg(this->channel));
            break;
         case ircCodes::RPL_WHOREPLY:
            //Create ID
            if(inCommand(&command[2],qts(this->channel).c_str())){
                userNick = command[6];
                if(userNick != this->ownNickAndStatus.nick){
                    userId = userNick + "!" + command[3] + "@" + command[4];
                    emit this->userOnline(getNickAndStatus(userNick),userId,
                                          this->channel);
                }
            }
            break;
        case ircCodes::RPL_ENDOFNAMES:

            break;
         default:
            break;
        }
    }else{ //Probably a user:
        if(command[0] == "PRIVMSG"){
            //Yep, it is!
            QStringList args = command;
            args.pop_front();
            QString channel = args[0];
            args.pop_front();

            emit this->chatReceived(channel, getNickAndStatus(sender), args.
join(" ").mid(1));
        }else if(command[0] == "NICK"){
            //Somebody changed nick!
            if(getNickAndStatus(sender).nick == this->
ownUser.getName()){

            }else{
                if(getNickAndStatusFromId(sender).nick ==
                        getNickAndStatus(command[1]).nick){
                    //Probably a status change:
                    emit this->userChangeStatus(
getNickAndStatusFromId(sender),
                                                getNickAndStatus(command[1]),
                            getId(sender));
                }else{
                    emit this->userChangeNick(
getNickAndStatusFromId(sender),
                                                getNickAndStatus(command[1]),
                            getId(sender));
                }
            }
        }else if(command[0] == "JOIN"){
            //Somebody joined, goody! :D
            if(getNickAndStatusFromId(sender).nick == this->
ownUser.getName()){
                //We have succesfully joined the channel
                //Yay, we can save our own ID.
                //Also, get a list of all the people in there
                this->send("WHO");
            }else{
                emit this->userOnline(getNickAndStatusFromId(sender),
sender,
                                      command[1].mid(1));
            }
        }else if(command[0] == "PART"){
            if(getNickAndStatusFromId(sender).nick == this->
ownUser.getName()){
                //Wait, what?
            }else{
                emit this->userOffline(getNickAndStatusFromId(sender),
sender,
                                       command[1]);
            }
        }
    }
}



void ircClient::sendChat(const char* msg, const char* channel){
    QString cmd = "PRIVMSG #" + QString(channel) + " :" + QString(msg);
```

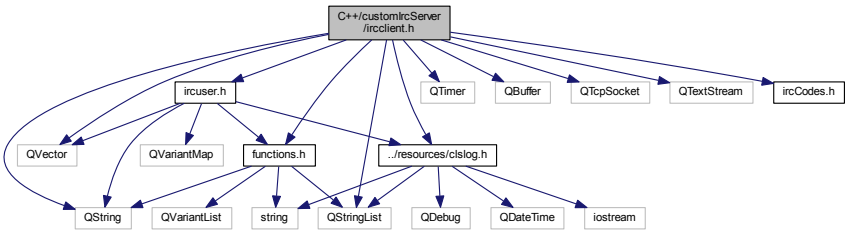
```
    this->send(qts(cmd).c_str());

    emit this->chatReceived(QString(channel),this->ownNickAndStatus,QString(msg));
}

void ircClient::sendChat(const char *msg){
    QString cmd = "PRIVMSG #" + this->channel + " :" + QString(msg);
    this->send(qts(cmd).c_str());
    emit this->chatReceived(QString(channel),this->ownNickAndStatus,QString(msg));
}

void ircClient::sendChat(const QString &msg){
    QString cmd = "PRIVMSG #" + this->channel + " :" + QString(msg);
    this->send(qts(cmd).c_str());
    emit this->chatReceived(QString(channel),this->ownNickAndStatus,QString(msg));
}

void ircClient::setUser(ircUser &user){
    this->ownUser = user;
}

nickAndStatus ircClient::getOwnNick() const{
    return this->ownNickAndStatus;
}

void ircClient::setOwnNick(const QString &newNick){
    QString cmd = "NICK " + newNick;
    this->send(cmd);
}


void ircClient::send(const QString data){
    //doLog(this->TAG, "Sending data: \t%s",data);
    this->log << "Sending data: \t" << data << endl;
    this->sock->write(qts(data).c_str());
    this->sock->write("\r\n");
    this->sock->flush();
}

void ircClient::readData(){
    qint64 bytes = this->buffer->write(this->sock->readAll());
    this->buffer->seek(this->buffer->pos() - bytes);
    // read only full lines, line by line
    while (this->buffer->canReadLine()){
        QString line = this->buffer->readLine();
        line = line.simplified();
        //doLog(this->TAG, "Data received: \t%s",qts(line).c_str());
        this->log << "Data received: \t" << line << endl;
        if(line.left(4) == QString("PING")){
            //Send a pong:
            QString toSend = "PONG" + line.mid(4);
            this->send(qts(toSend).c_str());
            //Previous one, put it somewhere
            timeBetweenPings = this->lastPing.elapsed();
            this->lastPing.start();
        }else if(line.left(5) == QString("ERROR")){
            //Handle.. once :P
        }else{
            line = line.mid(1);
            QStringList commandList = line.split(" ");
            QString sender = commandList.at(0);
            commandList.pop_front(); // Remove the sender

            this->handleCommand(sender,commandList);

        }
    }
}

void ircClient::connectTimeout(){
    if(this->connected == false && this->connecting == true){
        this->connecting = false;
        this->sock->abort();
        emit this->IRCConnectTimeOut(); //User can decide wether to reconnect immediately
    }
}

void ircClient::checkForPing(){
    if((timeBetweenPings - this->lastPing.elapsed()) < (PING_THRESHOLD*-1000) &&
            (timeBetweenPings > 0)){
        suppressDisconnectSignal = true;
        this->sock->abort();
        this->sock->disconnectFromHost();
```

```
        emit this->IRCDisconnected("Ping timeout");
        this->timeOutChecker->stop();
        timeBetweenPings = 0;


    }
}


}
```

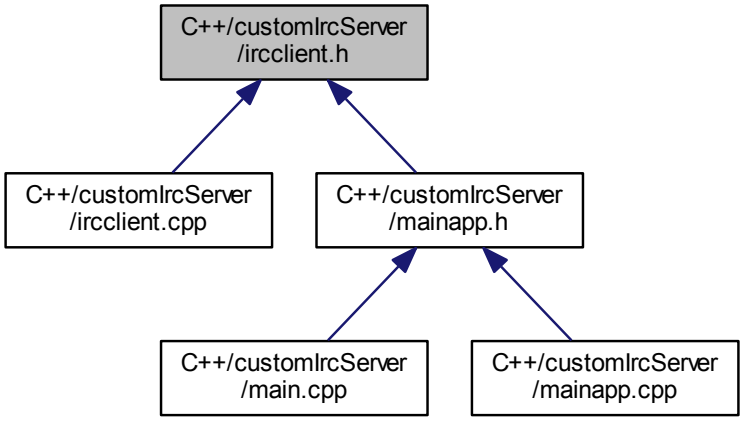
## 8.87 C++/customIrcServer/ircclient.h File Reference

```
#include <QVector>
#include <QTimer>
#include <QBuffer>
#include <QString>
#include <QTcpSocket>
#include <QStringList>
#include <QTextStream>
#include "ircCodes.h"
#include "functions.h"
#include "../resources/clslog.h"
#include "ircuser.h"
```

Include dependency graph for ircclient.h:



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




## Classes

- class server::ircClient

  *The ircClient class.*

## Namespaces

- namespace server

## Macros

- #define PORT 6667
- #define MAX_CONNECT_TIMEOUT_IN_MS 5000
- #define PING_THRESHOLD 50
- #define PING_CHECK_INTERVAL 5

### 8.87.1 Macro Definition Documentation

#### 8.87.1.1 #define MAX_CONNECT_TIMEOUT_IN_MS 5000

Max timeout for connecting, in MS

Definition at line 17 of file ircclient.h.

#### 8.87.1.2 #define PING_CHECK_INTERVAL 5

Definition at line 19 of file ircclient.h.

#### 8.87.1.3 #define PING_THRESHOLD 50

How long ago a ping may have been send

Definition at line 18 of file ircclient.h.

#### 8.87.1.4 #define PORT 6667

Standard IRC Port

Definition at line 16 of file ircclient.h.

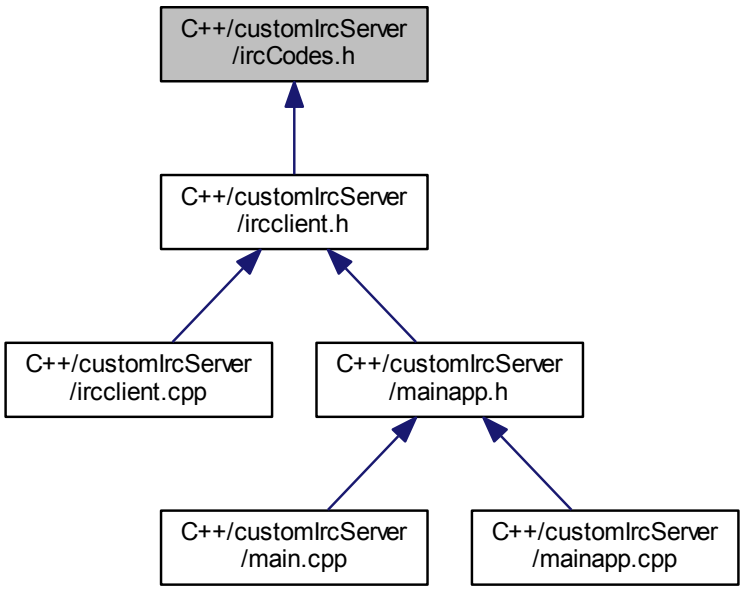
## 8.88 ircclient.h

```
00001 #ifndef IRCCLIENT_H
00002 #define IRCCLIENT_H
00003
00004 #include <QVector>
00005 #include <QTimer>
00006 #include <QBuffer>
00007 #include <QString>
00008 #include <QTcpSocket>
00009 #include <QStringList>
00010 #include <QTextStream>
00011 #include "ircCodes.h"
00012 #include "functions.h"
00013 #include "../resources/clslog.h"
00014 #include "ircuser.h"
00015
00016 #define PORT 6667
00017 #define MAX_CONNECT_TIMEOUT_IN_MS   5000
00018 #define PING_THRESHOLD  50
00019 #define PING_CHECK_INTERVAL 5
00020 namespace server {
00027 class ircClient : public QObject{
00028     Q_OBJECT
00029
00030 public:
00038     ircClient(const QString &hostname, const QString &channel, const QString &nick,
00039               const int port = PORT);
00040     ~ircClient();
00041
00045     void doConnect();
00046
00051     void setOwnUser(ircUser *ownUser);
00052
00058     void sendChat(const char *msg, const char *channel);
00063     void sendChat(const char *msg);
00068     void sendChat(const QString &msg);
00069
00074     void setUser(ircUser &user);
00079     nickAndStatus getOwnNick() const;
00080
00085     void setOwnNick(const QString &newNick);
00086 signals:
00090     void startupComplete();
00091
00098     void chatReceived(const QString channel, const nickAndStatus user,
00099                       const QString message);
00100
00107     void userOnline(const nickAndStatus nickName, const QString id,
00108                     const QString channel);
00109
00116     void userOffline(const nickAndStatus nickName, const QString id,
00117                      const QString channel);
00118
00126     void userChangeNick(const nickAndStatus oldNickName,
00127                         const nickAndStatus newNickName,
00128                         const QString id);
00136     void userChangeStatus(const nickAndStatus oldNick,
00137                           const nickAndStatus newNick,
00138                           const QString id);
00139
00143     void IRCConnectTimeOut();
00144
00148     void IRCConnected();
00149
00153     void IRCDisconnected(QString reason);
```

```
00154
00155 private:
00159     void init();
00160
00165     void send(const QString data);
00166
00172     void handleCommand(const QString &sender, const QStringList &command);
00173
00178     void sendPong(const QString packet);
00179
00180 public slots:
00181     void disconnected();
00182     void sltConnected();
00186     void readData();
00187     void connectTimeout();
00188
00192     void checkForPing();
00193 private:
00197     QString hostname;
00201     int port;
00205     QString channel;
00209     bool motdReceived;
00210
00214     QTcpSocket *sock;
00215
00219     QBuffer *buffer;
00220
00224     clsLog log;
00225
00226     nickAndStatus ownNickAndStatus;
00227     ircUser ownUser;
00228     bool connected;
00229     bool connecting;
00230
00231     QTimer *connectTimer;
00232     QTimer *timeOutChecker;
00233     QTime lastPing;
00234     int timeBetweenPings;
00235     bool suppressDisconnectSignal;
00236 };
00237 }
00238 #endif // IRCCLIENT_H
```

## 8.89 C++/customIrcServer/ircCodes.h File Reference

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




### Namespaces

- namespace ircCodes

### Enumerations

- enum ircCodes::ircCode {
  ircCodes::RPL_WELCOME = 1, ircCodes::RPL_YOURHOST = 2, ircCodes::RPL_CREATED = 3, ircCodes::RPL_MYINFO = 4,
  ircCodes::RPL_ISUPPORT = 5, ircCodes::RPL_SNOMASK = 8, ircCodes::RPL_STATMEMTOT = 9, ircCodes::RPL_BOUNCE = 10,
  ircCodes::RPL_STATMEM = 10, ircCodes::RPL_YOURCOOKIE = 14, ircCodes::RPL_YOURID = 42, ircCodes::RPL_SAVENICK = 43,
  ircCodes::RPL_ATTEMPTINGJUNC = 50, ircCodes::RPL_ATTEMPTINGREROUTE = 51, ircCodes::RPL_TRACELINK = 200, ircCodes::RPL_TRACECONNECTING = 201,
  ircCodes::RPL_TRACEHANDSHAKE = 202, ircCodes::RPL_TRACEUNKNOWN = 203, ircCodes::RPL_TRACEOPERATOR = 204, ircCodes::RPL_TRACEUSER = 205,
  ircCodes::RPL_TRACESERVER = 206, ircCodes::RPL_TRACESERVICE = 207, ircCodes::RPL_TRACENEWTYPE = 208, ircCodes::RPL_TRACECLASS = 209,
  ircCodes::RPL_TRACERECONNECT = 210, ircCodes::RPL_STATS = 210, ircCodes::RPL_STATSLINKINFO = 211, ircCodes::RPL_STATSCOMMANDS = 212,
  ircCodes::RPL_STATSCLINE = 213, ircCodes::RPL_STATSNLINE = 214, ircCodes::RPL_STATSILINE = 215, ircCodes::RPL_STATSKLINE = 216,
  ircCodes::RPL_STATSQLINE = 217, ircCodes::RPL_STATSYLINE = 218, ircCodes::RPL_ENDOFSTATS = 219, ircCodes::RPL_UMODEIS = 221,
  ircCodes::RPL_MODLIST = 222, ircCodes::RPL_SQLINE_NICK = 222, ircCodes::RPL_STATSZLINE = 225,

ircCodes::RPL_STATSCOUNT = 226,
ircCodes::RPL_SERVICEINFO = 231, ircCodes::RPL_ENDOFSERVICES = 232, ircCodes::RPL_SERVICE = 233, ircCodes::RPL_SERVLIST = 234,
ircCodes::RPL_SERVLISTEND = 235, ircCodes::RPL_STATSVERBOSE = 236, ircCodes::RPL_STATSENGINE = 237, ircCodes::RPL_STATSIAUTH = 239,
ircCodes::RPL_STATSVLINE = 240, ircCodes::RPL_STATSLLINE = 241, ircCodes::RPL_STATSUPTIME = 242, ircCodes::RPL_STATSOLINE = 243,
ircCodes::RPL_STATSHLINE = 244, ircCodes::RPL_STATSSLINE = 245, ircCodes::RPL_STATSPING = 246, ircCodes::RPL_STATSBLINE = 247,
ircCodes::RPL_STATSDEFINE = 248, ircCodes::RPL_STATSDEBUG = 249, ircCodes::RPL_STATSDLINE = 250, ircCodes::RPL_STATSCONN = 250,
ircCodes::RPL_LUSERCLIENT = 251, ircCodes::RPL_LUSEROP = 252, ircCodes::RPL_LUSERUNKNOWN = 253, ircCodes::RPL_LUSERCHANNELS = 254,
ircCodes::RPL_LUSERME = 255, ircCodes::RPL_ADMINME = 256, ircCodes::RPL_ADMINLOC1 = 257, ircCodes::RPL_ADMINLOC2 = 258,
ircCodes::RPL_ADMINEMAIL = 259, ircCodes::RPL_TRACELOG = 261, ircCodes::RPL_TRACEPING = 262, ircCodes::RPL_TRACEEND = 262,
ircCodes::RPL_TRYAGAIN = 263, ircCodes::RPL_LOCALUSERS = 265, ircCodes::RPL_GLOBALUSERS = 266, ircCodes::RPL_START_NETSTAT = 267,
ircCodes::RPL_NETSTAT = 268, ircCodes::RPL_END_NETSTAT = 269, ircCodes::RPL_PRIVS = 270, ircCodes::RPL_SILELIST = 271,
ircCodes::RPL_ENDOFSILELIST = 272, ircCodes::RPL_NOTIFY = 273, ircCodes::RPL_ENDNOTIFY = 274, ircCodes::RPL_STATSDELTA = 274,
ircCodes::RPL_VCHANEXIST = 276, ircCodes::RPL_VCHANLIST = 277, ircCodes::RPL_VCHANHELP = 278, ircCodes::RPL_GLIST = 280,
ircCodes::RPL_ENDOFGLIST = 281, ircCodes::RPL_ACCEPTLIST = 281, ircCodes::RPL_ENDOFACCEPT = 282, ircCodes::RPL_JUPELIST = 282,
ircCodes::RPL_ENDOFJUPELIST = 283, ircCodes::RPL_FEATURE = 284, ircCodes::RPL_GLIST_HASH = 285, ircCodes::RPL_CHANINFO_HANDLE = 285,
ircCodes::RPL_NEWHOSTIS = 285, ircCodes::RPL_CHANINFO_USERS = 286, ircCodes::RPL_CHKHEAD = 286, ircCodes::RPL_CHANINFO_CHOPS = 287,
ircCodes::RPL_CHANUSER = 287, ircCodes::RPL_CHANINFO_VOICES = 288, ircCodes::RPL_PATCHHEAD = 288, ircCodes::RPL_CHANINFO_AWAY = 289,
ircCodes::RPL_PATCHCON = 289, ircCodes::RPL_CHANINFO_OPERS = 290, ircCodes::RPL_HELPHDR = 290, ircCodes::RPL_DATASTR = 290,
ircCodes::RPL_CHANINFO_BANNED = 291, ircCodes::RPL_HELPOP = 291, ircCodes::RPL_ENDOFCHECK = 291, ircCodes::RPL_CHANINFO_BANS = 292,
ircCodes::RPL_HELPTLR = 292, ircCodes::RPL_CHANINFO_INVITE = 293, ircCodes::RPL_HELPHLP = 293, ircCodes::RPL_CHANINFO_INVITES = 294,
ircCodes::RPL_HELPFWD = 294, ircCodes::RPL_CHANINFO_KICK = 295, ircCodes::RPL_HELPIGN = 295, ircCodes::RPL_CHANINFO_KICKS = 296,
ircCodes::RPL_END_CHANINFO = 299, ircCodes::RPL_NONE = 300, ircCodes::RPL_AWAY = 301, ircCodes::RPL_USERHOST = 302,
ircCodes::RPL_ISON = 303, ircCodes::RPL_TEXT = 304, ircCodes::RPL_UNAWAY = 305, ircCodes::RPL_NOWAWAY = 306,
ircCodes::RPL_WHOISREGNICK = 307, ircCodes::RPL_SUSERHOST = 307, ircCodes::RPL_NOTIFYACTION = 308, ircCodes::RPL_WHOISADMIN = 308,
ircCodes::RPL_NICKTRACE = 309, ircCodes::RPL_WHOISSADMIN = 309, ircCodes::RPL_WHOISHELPER = 309, ircCodes::RPL_WHOISSVCMSG = 310,
ircCodes::RPL_WHOISHELPOP = 310, ircCodes::RPL_WHOISSERVICE = 310, ircCodes::RPL_WHOISUSER = 311, ircCodes::RPL_WHOISSERVER = 312,
ircCodes::RPL_WHOISOPERATOR = 313, ircCodes::RPL_WHOWASUSER = 314, ircCodes::RPL_ENDOFWHO = 315, ircCodes::RPL_WHOISCHANOP = 316,
ircCodes::RPL_WHOISIDLE = 317, ircCodes::RPL_ENDOFWHOIS = 318, ircCodes::RPL_WHOISCHANNELS = 319, ircCodes::RPL_WHOISVIRT = 320,
ircCodes::RPL_WHOIS_HIDDEN = 320, ircCodes::RPL_WHOISSPECIAL = 320, ircCodes::RPL_LISTSTART = 321, ircCodes::RPL_LIST = 322,
ircCodes::RPL_LISTEND = 323, ircCodes::RPL_CHANNELMODEIS = 324, ircCodes::RPL_UNIQOPIS =

325, ircCodes::RPL_CHANNELPASSIS = 325,
ircCodes::RPL_NOCHANPASS = 326, ircCodes::RPL_CHPASSUNKNOWN = 327, ircCodes::RPL_CHANNEL_URL = 328, ircCodes::RPL_CREATIONTIME = 329,
ircCodes::RPL_WHOWAS_TIME = 330, ircCodes::RPL_WHOISACCOUNT = 330, ircCodes::RPL_NOTOPIC = 331, ircCodes::RPL_TOPIC = 332,
ircCodes::RPL_TOPICWHOTIME = 333, ircCodes::RPL_LISTUSAGE = 334, ircCodes::RPL_COMMANDSYNTAX = 334, ircCodes::RPL_LISTSYNTAX = 334,
ircCodes::RPL_CHANPASSOK = 338, ircCodes::RPL_WHOISACTUALLY = 338, ircCodes::RPL_BADCHANPASS = 339, ircCodes::RPL_INVITING = 341,
ircCodes::RPL_SUMMONING = 342, ircCodes::RPL_INVITED = 345, ircCodes::RPL_INVITELIST = 346, ircCodes::RPL_ENDOFINVITELIST = 347,
ircCodes::RPL_EXCEPTLIST = 348, ircCodes::RPL_ENDOFEXCEPTLIST = 349, ircCodes::RPL_VERSION = 351, ircCodes::RPL_WHOREPLY = 352,
ircCodes::RPL_NAMREPLY = 353, ircCodes::RPL_WHOSPCRPL = 354, ircCodes::RPL_NAMREPLY_ = 355, ircCodes::RPL_KILLDONE = 361,
ircCodes::RPL_CLOSING = 362, ircCodes::RPL_CLOSEEND = 363, ircCodes::RPL_LINKS = 364, ircCodes::RPL_ENDOFLINKS = 365,
ircCodes::RPL_ENDOFNAMES = 366, ircCodes::RPL_BANLIST = 367, ircCodes::RPL_ENDOFBANLIST = 368, ircCodes::RPL_ENDOFWHOWAS = 369,
ircCodes::RPL_INFO = 371, ircCodes::RPL_MOTD = 372, ircCodes::RPL_INFOSTART = 373, ircCodes::RPL_ENDOFINFO = 374,
ircCodes::RPL_MOTDSTART = 375, ircCodes::RPL_ENDOFMOTD = 376, ircCodes::RPL_KICKEXPIRED = 377, ircCodes::RPL_SPAM = 377,
ircCodes::RPL_BANEXPIRED = 378, ircCodes::RPL_WHOISHOST = 378, ircCodes::RPL_KICKLINKED = 379, ircCodes::RPL_WHOISMODES = 379,
ircCodes::RPL_BANLINKED = 380, ircCodes::RPL_YOURHELPER = 380, ircCodes::RPL_YOUREOPER = 381, ircCodes::RPL_REHASHING = 382,
ircCodes::RPL_YOURESERVICE = 383, ircCodes::RPL_MYPORTIS = 384, ircCodes::RPL_NOTOPERANYMORE = 385, ircCodes::RPL_QLIST = 386,
ircCodes::RPL_IRCOPS = 386, ircCodes::RPL_ENDOFQLIST = 387, ircCodes::RPL_ENDOFIRCOPS = 387, ircCodes::RPL_ALIST = 388,
ircCodes::RPL_ENDOFALIST = 389, ircCodes::RPL_TIME = 391, ircCodes::RPL_USERSSTART = 392, ircCodes::RPL_USERS = 393,
ircCodes::RPL_ENDOFUSERS = 394, ircCodes::RPL_NOUSERS = 395, ircCodes::RPL_HOSTHIDDEN = 396, ircCodes::ERR_UNKNOWNERROR = 400,
ircCodes::ERR_NOSUCHNICK = 401, ircCodes::ERR_NOSUCHSERVER = 402, ircCodes::ERR_NOSUCHCHANNEL = 403, ircCodes::ERR_CANNOTSENDTOCHAN = 404,
ircCodes::ERR_TOOMANYCHANNELS = 405, ircCodes::ERR_WASNOSUCHNICK = 406, ircCodes::ERR_TOOMANYTARGETS = 407, ircCodes::ERR_NOSUCHSERVICE = 408,
ircCodes::ERR_NOCOLORSONCHAN = 408, ircCodes::ERR_NOORIGIN = 409, ircCodes::ERR_NORECIPIENT = 411, ircCodes::ERR_NOTEXTTOSEND = 412,
ircCodes::ERR_NOTOPLEVEL = 413, ircCodes::ERR_WILDTOPLEVEL = 414, ircCodes::ERR_BADMASK = 415, ircCodes::ERR_TOOMANYMATCHES = 416,
ircCodes::ERR_QUERYTOOLONG = 416, ircCodes::ERR_LENGTHTRUNCATED = 419, ircCodes::ERR_UNKNOWNCOMMAND = 421, ircCodes::ERR_NOMOTD = 422,
ircCodes::ERR_NOADMININFO = 423, ircCodes::ERR_FILEERROR = 424, ircCodes::ERR_NOOPERMOTD = 425, ircCodes::ERR_TOOMANYAWAY = 429,
ircCodes::ERR_EVENTNICKCHANGE = 430, ircCodes::ERR_NONICKNAMEGIVEN = 431, ircCodes::ERR_ERRONEUSNICKNAME = 432, ircCodes::ERR_NICKNAMEINUSE = 433,
ircCodes::ERR_SERVICENAMEINUSE = 434, ircCodes::ERR_NORULES = 434, ircCodes::ERR_SERVICECONFUSED = 435, ircCodes::ERR_BANONCHAN = 435,
ircCodes::ERR_NICKCOLLISION = 436, ircCodes::ERR_UNAVAILRESOURCE = 437, ircCodes::ERR_BANNICKCHANGE = 437, ircCodes::ERR_NICKTOOFAST = 438,
ircCodes::ERR_DEAD = 438, ircCodes::ERR_TARGETTOOFAST = 439, ircCodes::ERR_SERVICESDOWN = 440, ircCodes::ERR_USERNOTINCHANNEL = 441,
ircCodes::ERR_NOTONCHANNEL = 442, ircCodes::ERR_USERONCHANNEL = 443, ircCodes::ERR_NOLOGIN = 444, ircCodes::ERR_SUMMONDISABLED = 445,
ircCodes::ERR_USERSDISABLED = 446, ircCodes::ERR_NONICKCHANGE = 447, ircCodes::ERR_NOTI-

MPLEMENTED = 449, ircCodes::ERR_NOTREGISTERED = 451,
ircCodes::ERR_IDCOLLISION = 452, ircCodes::ERR_NICKLOST = 453, ircCodes::ERR_HOSTILENAME = 455, ircCodes::ERR_ACCEPTFULL = 456,
ircCodes::ERR_ACCEPTEXIST = 457, ircCodes::ERR_ACCEPTNOT = 458, ircCodes::ERR_NOHIDING = 459, ircCodes::ERR_NOTFORHALFOPS = 460,
ircCodes::ERR_NEEDMOREPARAMS = 461, ircCodes::ERR_ALREADYREGISTERED = 462, ircCodes::ERR_NOPERMFORHOST = 463, ircCodes::ERR_PASSWDMISMATCH = 464,
ircCodes::ERR_YOUREBANNEDCREEP = 465, ircCodes::ERR_YOUWILLBEBANNED = 466, ircCodes::ERR_KEYSET = 467, ircCodes::ERR_INVALIDUSERNAME = 468,
ircCodes::ERR_ONLYSERVERSCANCHANGE = 468, ircCodes::ERR_LINKSET = 469, ircCodes::ERR_LINKCHANNEL = 470, ircCodes::ERR_KICKEDFROMCHAN = 470,
ircCodes::ERR_CHANNELISFULL = 471, ircCodes::ERR_UNKNOWNMODE = 472, ircCodes::ERR_INVITEONLYCHAN = 473, ircCodes::ERR_BANNEDFROMCHAN = 474,
ircCodes::ERR_BADCHANNELKEY = 475, ircCodes::ERR_BADCHANMASK = 476, ircCodes::ERR_NOCHANMODES = 477, ircCodes::ERR_NEEDREGGEDNICK = 477,
ircCodes::ERR_BANLISTFULL = 478, ircCodes::ERR_BADCHANNAME = 479, ircCodes::ERR_LINKFAIL = 479, ircCodes::ERR_NOULINE = 480,
ircCodes::ERR_CANNOTKNOCK = 480, ircCodes::ERR_NOPRIVILEGES = 481, ircCodes::ERR_CHANOPRIVSNEEDED = 482, ircCodes::ERR_CANTKILLSERVER = 483,
ircCodes::ERR_RESTRICTED = 484, ircCodes::ERR_ISCHANSERVICE = 484, ircCodes::ERR_DESYNC = 484, ircCodes::ERR_ATTACKDENY = 484,
ircCodes::ERR_UNIQOPRIVSNEEDED = 485, ircCodes::ERR_KILLDENY = 485, ircCodes::ERR_CANTKICKADMIN = 485, ircCodes::ERR_ISREALSERVICE = 485,
ircCodes::ERR_NONONREG = 486, ircCodes::ERR_HTMDISABLED = 486, ircCodes::ERR_ACCOUNTONLY = 486, ircCodes::ERR_CHANTOORECENT = 487,
ircCodes::ERR_MSGSERVICES = 487, ircCodes::ERR_TSLESSCHAN = 488, ircCodes::ERR_VOICENEEDED = 489, ircCodes::ERR_SECUREONLYCHAN = 489,
ircCodes::ERR_NOOPERHOST = 491, ircCodes::ERR_NOSERVICEHOST = 492, ircCodes::ERR_NOFEATURE = 493, ircCodes::ERR_BADFEATURE = 494,
ircCodes::ERR_BADLOGTYPE = 495, ircCodes::ERR_BADLOGSYS = 496, ircCodes::ERR_BADLOGVALUE = 497, ircCodes::ERR_ISOPERLCHAN = 498,
ircCodes::ERR_CHANOWNPRIVNEEDED = 499, ircCodes::ERR_UMODEUNKNOWNFLAG = 501, ircCodes::ERR_USERSDONTMATCH = 502, ircCodes::ERR_GHOSTEDCLIENT = 503,
ircCodes::ERR_VWORLDWARN = 503, ircCodes::ERR_USERNOTONSERV = 504, ircCodes::ERR_SILELISTFULL = 511, ircCodes::ERR_TOOMANYWATCH = 512,
ircCodes::ERR_BADPING = 513, ircCodes::ERR_INVALID_ERROR = 514, ircCodes::ERR_TOOMANYDCC = 514, ircCodes::ERR_BADEXPIRE = 515,
ircCodes::ERR_DONTCHEAT = 516, ircCodes::ERR_DISABLED = 517, ircCodes::ERR_NOINVITE = 518, ircCodes::ERR_LONGMASK = 518,
ircCodes::ERR_ADMONLY = 519, ircCodes::ERR_TOOMANYUSERS = 519, ircCodes::ERR_OPERONLY = 520, ircCodes::ERR_MASKTOOWIDE = 520,
ircCodes::ERR_WHOTRUNC = 520, ircCodes::ERR_LISTSYNTAX = 521, ircCodes::ERR_WHOSYNTAX = 522, ircCodes::ERR_WHOLIMEXCEED = 523,
ircCodes::ERR_QUARANTINED = 524, ircCodes::ERR_OPERSPVERIFY = 524, ircCodes::ERR_REMOTEPFX = 525, ircCodes::ERR_PFXUNROUTABLE = 526,
ircCodes::ERR_BADHOSTMASK = 550, ircCodes::ERR_HOSTUNAVAIL = 551, ircCodes::ERR_USINGSLINE = 552, ircCodes::ERR_STATSSLINE = 553,
ircCodes::RPL_LOGON = 600, ircCodes::RPL_LOGOFF = 601, ircCodes::RPL_WATCHOFF = 602, ircCodes::RPL_WATCHSTAT = 603,
ircCodes::RPL_NOWON = 604, ircCodes::RPL_NOWOFF = 605, ircCodes::RPL_WATCHLIST = 606, ircCodes::RPL_ENDOFWATCHLIST = 607,
ircCodes::RPL_WATCHCLEAR = 608, ircCodes::RPL_ISOPER = 610, ircCodes::RPL_ISLOCOP = 611, ircCodes::RPL_ISNOTOPER = 612,
ircCodes::RPL_ENDOFISOPER = 613, ircCodes::RPL_DCCSTATUS = 617, ircCodes::RPL_DCCLIST = 618, ircCodes::RPL_ENDOFDCCLIST = 619,
ircCodes::RPL_WHOWASHOST = 619, ircCodes::RPL_DCCINFO = 620, ircCodes::RPL_ENDOFO = 626, ircCodes::RPL_SETTINGS = 630,
ircCodes::RPL_ENDOFSETTINGS = 631, ircCodes::RPL_DUMPING = 640, ircCodes::RPL_DUMPRPL =

641, ircCodes::RPL_EODUMP = 642,
ircCodes::RPL_TRACEROUTE_HOP = 660, ircCodes::RPL_TRACEROUTE_START = 661, ircCodes::RPL_MODECHANGEWARN = 662, ircCodes::RPL_CHANREDIR = 663,
ircCodes::RPL_SERVMODEIS = 664, ircCodes::RPL_OTHERUMODEIS = 665, ircCodes::RPL_ENDOF_GENERIC = 666, ircCodes::RPL_WHOWASDETAILS = 670,
ircCodes::RPL_WHOISSECURE = 671, ircCodes::RPL_UNKNOWNMODES = 672, ircCodes::RPL_CANNOTSETMODES = 673, ircCodes::RPL_LUSERSTAFF = 678,
ircCodes::RPL_TIMEONSERVERIS = 679, ircCodes::RPL_NETWORKS = 682, ircCodes::RPL_YOURLANGUAGEIS = 687, ircCodes::RPL_LANGUAGE = 688,
ircCodes::RPL_WHOISSTAFF = 689, ircCodes::RPL_WHOISLANGUAGE = 690, ircCodes::RPL_HELPSTART = 704, ircCodes::RPL_HELPTXT = 705,
ircCodes::RPL_ENDOFHELP = 706, ircCodes::RPL_ETRACEFULL = 708, ircCodes::RPL_ETRACE = 709, ircCodes::RPL_KNOCK = 710,
ircCodes::RPL_KNOCKDLVR = 711, ircCodes::ERR_TOOMANYKNOCK = 712, ircCodes::ERR_CHANOPEN = 713, ircCodes::ERR_KNOCKONCHAN = 714,
ircCodes::ERR_KNOCKDISABLED = 715, ircCodes::RPL_TARGUMODEG = 716, ircCodes::RPL_TARGNOTIFY = 717, ircCodes::RPL_UMODEGMSG = 718,
ircCodes::RPL_ENDOFOMOTD = 722, ircCodes::ERR_NOPRIVS = 723, ircCodes::RPL_TESTMARK = 724, ircCodes::RPL_TESTLINE = 725,
ircCodes::RPL_NOTESTLINE = 726, ircCodes::RPL_XINFO = 771, ircCodes::RPL_XINFOSTART = 773, ircCodes::RPL_XINFOEND = 774,
ircCodes::ERR_CANNOTDOCOMMAND = 972, ircCodes::ERR_CANNOTCHANGEUMODE = 973, ircCodes::ERR_CANNOTCHANGECHANMODE = 974, ircCodes::ERR_CANNOTCHANGESERVERMODE = 975,
ircCodes::ERR_CANNOTSENDTONICK = 976, ircCodes::ERR_UNKNOWNSERVERMODE = 977, ircCodes::ERR_SERVERMODELOCK = 979, ircCodes::ERR_BADCHARENCODING = 980,
ircCodes::ERR_TOOMANYLANGUAGES = 981, ircCodes::ERR_NOLANGUAGE = 982, ircCodes::ERR_TEXTTOOSHORT = 983, ircCodes::ERR_NUMERIC_ERR = 999 }

## 8.90 ircCodes.h

```
#ifndef IRCCODES_H
#define IRCCODES_H

namespace ircCodes {
enum ircCode {
  RPL_WELCOME = 1, RPL_YOURHOST = 2, RPL_CREATED = 3, RPL_MYINFO = 4,
  RPL_ISUPPORT = 5, RPL_SNOMASK = 8, RPL_STATMEMTOT = 9, RPL_BOUNCE = 10,
  RPL_STATMEM = 10, RPL_YOURCOOKIE = 14, RPL_YOURID = 42, RPL_SAVENICK = 43,
  RPL_ATTEMPTINGJUNC = 50, RPL_ATTEMPTINGREROUTE = 51, RPL_TRACELINK = 200, RPL_TRACECONNECTING = 201,
  RPL_TRACEHANDSHAKE = 202, RPL_TRACEUNKNOWN = 203, RPL_TRACEOPERATOR = 204, RPL_TRACEUSER = 205,
  RPL_TRACESERVER = 206, RPL_TRACESERVICE = 207, RPL_TRACENEWTYPE = 208, RPL_TRACECLASS = 209,
  RPL_TRACERECONNECT = 210, RPL_STATS = 210, RPL_STATSLINKINFO = 211, RPL_STATSCOMMANDS = 212,
  RPL_STATSCLINE = 213, RPL_STATSNLINE = 214, RPL_STATSILINE = 215, RPL_STATSKLINE = 216,
  RPL_STATSQLINE = 217, RPL_STATSYLINE = 218, RPL_ENDOFSTATS = 219, RPL_UMODEIS = 221,
  RPL_MODLIST = 222, RPL_SQLINE_NICK = 222, RPL_STATSZLINE = 225, RPL_STATSCOUNT = 226,
  RPL_SERVICEINFO = 231, RPL_ENDOFSERVICES = 232, RPL_SERVICE = 233, RPL_SERVLIST = 234,
  RPL_SERVLISTEND = 235, RPL_STATSVERBOSE = 236, RPL_STATSENGINE = 237, RPL_STATSIAUTH = 239,
  RPL_STATSVLINE = 240, RPL_STATSLLINE = 241, RPL_STATSUPTIME = 242, RPL_STATSOLINE = 243,
  RPL_STATSHLINE = 244, RPL_STATSSLINE = 245, RPL_STATSPING = 246, RPL_STATSBLINE = 247,
  RPL_STATSDEFINE = 248, RPL_STATSDEBUG = 249, RPL_STATSDLINE = 250, RPL_STATSCONN = 250,
  RPL_LUSERCLIENT = 251, RPL_LUSEROP = 252, RPL_LUSERUNKNOWN = 253, RPL_LUSERCHANNELS = 254,
  RPL_LUSERME = 255, RPL_ADMINME = 256, RPL_ADMINLOC1 = 257, RPL_ADMINLOC2 = 258,
```

```
00023   RPL_ADMINEMAIL = 259, RPL_TRACELOG = 261,
      RPL_TRACEPING = 262, RPL_TRACEEND = 262,
00024   RPL_TRYAGAIN = 263, RPL_LOCALUSERS = 265,
      RPL_GLOBALUSERS = 266, RPL_START_NETSTAT = 267,
00025   RPL_NETSTAT = 268, RPL_END_NETSTAT = 269, RPL_PRIVS = 270,
      RPL_SILELIST = 271,
00026   RPL_ENDOFSILELIST = 272, RPL_NOTIFY = 273,
      RPL_ENDNOTIFY = 274, RPL_STATSDELTA = 274,
00027   RPL_VCHANEXIST = 276, RPL_VCHANLIST = 277,
      RPL_VCHANHELP = 278, RPL_GLIST = 280,
00028   RPL_ENDOFGLIST = 281, RPL_ACCEPTLIST = 281,
      RPL_ENDOFACCEPT = 282, RPL_JUPELIST = 282,
00029   RPL_ENDOFJUPELIST = 283, RPL_FEATURE = 284,
      RPL_GLIST_HASH = 285, RPL_CHANINFO_HANDLE = 285,
00030   RPL_NEWHOSTIS = 285, RPL_CHANINFO_USERS = 286,
      RPL_CHKHEAD = 286, RPL_CHANINFO_CHOPS = 287,
00031   RPL_CHANUSER = 287, RPL_CHANINFO_VOICES = 288,
      RPL_PATCHHEAD = 288, RPL_CHANINFO_AWAY = 289,
00032   RPL_PATCHCON = 289, RPL_CHANINFO_OPERS = 290,
      RPL_HELPHDR = 290, RPL_DATASTR = 290,
00033   RPL_CHANINFO_BANNED = 291, RPL_HELPOP = 291,
      RPL_ENDOFCHECK = 291, RPL_CHANINFO_BANS = 292,
00034   RPL_HELPTLR = 292, RPL_CHANINFO_INVITE = 293,
      RPL_HELPHLP = 293, RPL_CHANINFO_INVITES = 294,
00035   RPL_HELPFWD = 294, RPL_CHANINFO_KICK = 295,
      RPL_HELPIGN = 295, RPL_CHANINFO_KICKS = 296,
00036   RPL_END_CHANINFO = 299, RPL_NONE = 300, RPL_AWAY = 301,
      RPL_USERHOST = 302,
00037   RPL_ISON = 303, RPL_TEXT = 304, RPL_UNAWAY = 305,
      RPL_NOWAWAY = 306,
00038   RPL_WHOISREGNICK = 307, RPL_SUSERHOST = 307,
      RPL_NOTIFYACTION = 308, RPL_WHOISADMIN = 308,
00039   RPL_NICKTRACE = 309, RPL_WHOISSADMIN = 309,
      RPL_WHOISHELPER = 309, RPL_WHOISSVCMSG = 310,
00040   RPL_WHOISHELPOP = 310, RPL_WHOISSERVICE = 310,
      RPL_WHOISUSER = 311, RPL_WHOISSERVER = 312,
00041   RPL_WHOISOPERATOR = 313, RPL_WHOWASUSER = 314,
      RPL_ENDOFWHO = 315, RPL_WHOISCHANOP = 316,
00042   RPL_WHOISIDLE = 317, RPL_ENDOFWHOIS = 318,
      RPL_WHOISCHANNELS = 319, RPL_WHOISVIRT = 320,
00043   RPL_WHOIS_HIDDEN = 320, RPL_WHOISSPECIAL = 320,
      RPL_LISTSTART = 321, RPL_LIST = 322,
00044   RPL_LISTEND = 323, RPL_CHANNELMODEIS = 324,
      RPL_UNIQOPIS = 325, RPL_CHANNELPASSIS = 325,
00045   RPL_NOCHANPASS = 326, RPL_CHPASSUNKNOWN = 327,
      RPL_CHANNEL_URL = 328, RPL_CREATIONTIME = 329,
00046   RPL_WHOWAS_TIME = 330, RPL_WHOISACCOUNT = 330,
      RPL_NOTOPIC = 331, RPL_TOPIC = 332,
00047   RPL_TOPICWHOTIME = 333, RPL_LISTUSAGE = 334,
      RPL_COMMANDSYNTAX = 334, RPL_LISTSYNTAX = 334,
00048   RPL_CHANPASSOK = 338, RPL_WHOISACTUALLY = 338,
      RPL_BADCHANPASS = 339, RPL_INVITING = 341,
00049   RPL_SUMMONING = 342, RPL_INVITED = 345, RPL_INVITELIST = 346,
      RPL_ENDOFINVITELIST = 347,
00050   RPL_EXCEPTLIST = 348, RPL_ENDOFEXCEPTLIST = 349,
      RPL_VERSION = 351, RPL_WHOREPLY = 352,
00051   RPL_NAMREPLY = 353, RPL_WHOSPCRPL = 354,
      RPL_NAMREPLY_ = 355, RPL_KILLDONE = 361,
00052   RPL_CLOSING = 362, RPL_CLOSEEND = 363, RPL_LINKS = 364,
      RPL_ENDOFLINKS = 365,
00053   RPL_ENDOFNAMES = 366, RPL_BANLIST = 367,
      RPL_ENDOFBANLIST = 368, RPL_ENDOFWHOWAS = 369,
00054   RPL_INFO = 371, RPL_MOTD = 372, RPL_INFOSTART = 373,
      RPL_ENDOFINFO = 374,
00055   RPL_MOTDSTART = 375, RPL_ENDOFMOTD = 376,
      RPL_KICKEXPIRED = 377, RPL_SPAM = 377,
00056   RPL_BANEXPIRED = 378, RPL_WHOISHOST = 378,
      RPL_KICKLINKED = 379, RPL_WHOISMODES = 379,
00057   RPL_BANLINKED = 380, RPL_YOURHELPER = 380,
      RPL_YOUREOPER = 381, RPL_REHASHING = 382,
00058   RPL_YOURESERVICE = 383, RPL_MYPORTIS = 384,
      RPL_NOTOPERANYMORE = 385, RPL_QLIST = 386,
00059   RPL_IRCOPS = 386, RPL_ENDOFQLIST = 387,
      RPL_ENDOFIRCOPS = 387, RPL_ALIST = 388,
00060   RPL_ENDOFALIST = 389, RPL_TIME = 391, RPL_USERSSTART = 392,
      RPL_USERS = 393,
00061   RPL_ENDOFUSERS = 394, RPL_NOUSERS = 395,
      RPL_HOSTHIDDEN = 396, ERR_UNKNOWNERROR = 400,
00062   ERR_NOSUCHNICK = 401, ERR_NOSUCHSERVER = 402,
      ERR_NOSUCHCHANNEL = 403, ERR_CANNOTSENDTOCHAN = 404,
00063   ERR_TOOMANYCHANNELS = 405, ERR_WASNOSUCHNICK = 406,
      ERR_TOOMANYTARGETS = 407, ERR_NOSUCHSERVICE = 408,
00064   ERR_NOCOLORSONCHAN = 408, ERR_NOORIGIN = 409,
      ERR_NORECIPIENT = 411, ERR_NOTEXTTOSEND = 412,
00065   ERR_NOTOPLEVEL = 413, ERR_WILDTOPLEVEL = 414,
      ERR_BADMASK = 415, ERR_TOOMANYMATCHES = 416,
00066   ERR_QUERYTOOLONG = 416, ERR_LENGTHTRUNCATED = 419,
```

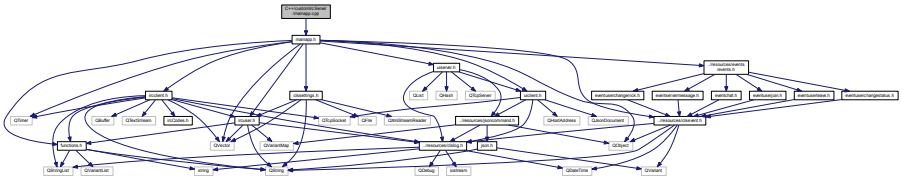
```
      ERR_UNKNOWNCOMMAND = 421, ERR_NOMOTD = 422,
00067   ERR_NOADMININFO = 423, ERR_FILEERROR = 424,
      ERR_NOOPERMOTD = 425, ERR_TOOMANYAWAY = 429,
00068   ERR_EVENTNICKCHANGE = 430, ERR_NONICKNAMEGIVEN = 431,
      ERR_ERRONEUSNICKNAME = 432, ERR_NICKNAMEINUSE = 433,
00069   ERR_SERVICENAMEINUSE = 434, ERR_NORULES = 434,
      ERR_SERVICECONFUSED = 435, ERR_BANONCHAN = 435,
00070   ERR_NICKCOLLISION = 436, ERR_UNAVAILRESOURCE = 437,
      ERR_BANNICKCHANGE = 437, ERR_NICKTOOFAST = 438,
00071   ERR_DEAD = 438, ERR_TARGETTOOFAST = 439,
      ERR_SERVICESDOWN = 440, ERR_USERNOTINCHANNEL = 441,
00072   ERR_NOTONCHANNEL = 442, ERR_USERONCHANNEL = 443,
      ERR_NOLOGIN = 444, ERR_SUMMONDISABLED = 445,
00073   ERR_USERSDISABLED = 446, ERR_NONICKCHANGE = 447,
      ERR_NOTIMPLEMENTED = 449, ERR_NOTREGISTERED = 451,
00074   ERR_IDCOLLISION = 452, ERR_NICKLOST = 453,
      ERR_HOSTILENAME = 455, ERR_ACCEPTFULL = 456,
00075   ERR_ACCEPTEXIST = 457, ERR_ACCEPTNOT = 458,
      ERR_NOHIDING = 459, ERR_NOTFORHALFOPS = 460,
00076   ERR_NEEDMOREPARAMS = 461, ERR_ALREADYREGISTERED = 462,
      ERR_NOPERMFORHOST = 463, ERR_PASSWDMISMATCH = 464,
00077   ERR_YOUREBANNEDCREEP = 465, ERR_YOUWILLBEBANNED = 466,
      ERR_KEYSET = 467, ERR_INVALIDUSERNAME = 468,
00078   ERR_ONLYSERVERSCANCHANGE = 468, ERR_LINKSET = 469,
      ERR_LINKCHANNEL = 470, ERR_KICKEDFROMCHAN = 470,
00079   ERR_CHANNELISFULL = 471, ERR_UNKNOWNMODE = 472,
      ERR_INVITEONLYCHAN = 473, ERR_BANNEDFROMCHAN = 474,
00080   ERR_BADCHANNELKEY = 475, ERR_BADCHANMASK = 476,
      ERR_NOCHANMODES = 477, ERR_NEEDREGGEDNICK = 477,
00081   ERR_BANLISTFULL = 478, ERR_BADCHANNAME = 479,
      ERR_LINKFAIL = 479, ERR_NOULINE = 480,
00082   ERR_CANNOTKNOCK = 480, ERR_NOPRIVILEGES = 481,
      ERR_CHANOPRIVSNEEDED = 482, ERR_CANTKILLSERVER = 483,
00083   ERR_RESTRICTED = 484, ERR_ISCHANSERVICE = 484,
      ERR_DESYNC = 484, ERR_ATTACKDENY = 484,
00084   ERR_UNIQOPRIVSNEEDED = 485, ERR_KILLDENY = 485,
      ERR_CANTKICKADMIN = 485, ERR_ISREALSERVICE = 485,
00085   ERR_NONONREG = 486, ERR_HTMDISABLED = 486,
      ERR_ACCOUNTONLY = 486, ERR_CHANTOORECENT = 487,
00086   ERR_MSGSERVICES = 487, ERR_TSLESSCHAN = 488,
      ERR_VOICENEEDED = 489, ERR_SECUREONLYCHAN = 489,
00087   ERR_NOOPERHOST = 491, ERR_NOSERVICEHOST = 492,
      ERR_NOFEATURE = 493, ERR_BADFEATURE = 494,
00088   ERR_BADLOGTYPE = 495, ERR_BADLOGSYS = 496,
      ERR_BADLOGVALUE = 497, ERR_ISOPERLCHAN = 498,
00089   ERR_CHANOWNPRIVNEEDED = 499, ERR_UMODEUNKNOWNFLAG = 501,
      ERR_USERSDONTMATCH = 502, ERR_GHOSTEDCLIENT = 503,
00090   ERR_VWORLDWARN = 503, ERR_USERNOTONSERV = 504,
      ERR_SILELISTFULL = 511, ERR_TOOMANYWATCH = 512,
00091   ERR_BADPING = 513, ERR_INVALID_ERROR = 514,
      ERR_TOOMANYDCC = 514, ERR_BADEXPIRE = 515,
00092   ERR_DONTCHEAT = 516, ERR_DISABLED = 517, ERR_NOINVITE = 518,
      ERR_LONGMASK = 518,
00093   ERR_ADMONLY = 519, ERR_TOOMANYUSERS = 519,
      ERR_OPERONLY = 520, ERR_MASKTOOWIDE = 520,
00094   ERR_WHOTRUNC = 520, ERR_LISTSYNTAX = 521,
      ERR_WHOSYNTAX = 522, ERR_WHOLIMEXCEED = 523,
00095   ERR_QUARANTINED = 524, ERR_OPERSPVERIFY = 524,
      ERR_REMOTEPFX = 525, ERR_PFXUNROUTABLE = 526,
00096   ERR_BADHOSTMASK = 550, ERR_HOSTUNAVAIL = 551,
      ERR_USINGSLINE = 552, ERR_STATSSLINE = 553,
00097   RPL_LOGON = 600, RPL_LOGOFF = 601, RPL_WATCHOFF = 602,
      RPL_WATCHSTAT = 603,
00098   RPL_NOWON = 604, RPL_NOWOFF = 605, RPL_WATCHLIST = 606,
      RPL_ENDOFWATCHLIST = 607,
00099   RPL_WATCHCLEAR = 608, RPL_ISOPER = 610, RPL_ISLOCOP = 611,
      RPL_ISNOTOPER = 612,
00100   RPL_ENDOFISOPER = 613, RPL_DCCSTATUS = 617,
      RPL_DCCLIST = 618, RPL_ENDOFDCCLIST = 619,
00101   RPL_WHOWASHOST = 619, RPL_DCCINFO = 620, RPL_ENDOFO = 626,
      RPL_SETTINGS = 630,
00102   RPL_ENDOFSETTINGS = 631, RPL_DUMPING = 640,
      RPL_DUMPRPL = 641, RPL_EODUMP = 642,
00103   RPL_TRACEROUTE_HOP = 660, RPL_TRACEROUTE_START = 661,
      RPL_MODECHANGEWARN = 662, RPL_CHANREDIR = 663,
00104   RPL_SERVMODEIS = 664, RPL_OTHERUMODEIS = 665,
      RPL_ENDOF_GENERIC = 666, RPL_WHOWASDETAILS = 670,
00105   RPL_WHOISSECURE = 671, RPL_UNKNOWNMODES = 672,
      RPL_CANNOTSETMODES = 673, RPL_LUSERSTAFF = 678,
00106   RPL_TIMEONSERVERIS = 679, RPL_NETWORKS = 682,
      RPL_YOURLANGUAGEIS = 687, RPL_LANGUAGE = 688,
00107   RPL_WHOISSTAFF = 689, RPL_WHOISLANGUAGE = 690,
      RPL_HELPSTART = 704, RPL_HELPTXT = 705,
00108   RPL_ENDOFHELP = 706, RPL_ETRACEFULL = 708,
      RPL_ETRACE = 709, RPL_KNOCK = 710,
00109   RPL_KNOCKDLVR = 711, ERR_TOOMANYKNOCK = 712,
      ERR_CHANOPEN = 713, ERR_KNOCKONCHAN = 714,
```

```
  ERR_KNOCKDISABLED = 715, RPL_TARGUMODEG = 716, RPL_TARGNOTIFY = 717, RPL_UMODEGMSG = 718,
  RPL_ENDOFOMOTD = 722, ERR_NOPRIVS = 723, RPL_TESTMARK = 724, RPL_TESTLINE = 725,
  RPL_NOTESTLINE = 726, RPL_XINFO = 771, RPL_XINFOSTART = 773, RPL_XINFOEND = 774,
  ERR_CANNOTDOCOMMAND = 972, ERR_CANNOTCHANGEUMODE = 973, ERR_CANNOTCHANGECHANMODE = 974,
  ERR_CANNOTCHANGESERVERMODE = 975,
  ERR_CANNOTSENDTONICK = 976, ERR_UNKNOWNSERVERMODE = 977, ERR_SERVERMODELOCK = 979, ERR_BADCHARENCODING = 980,
  ERR_TOOMANYLANGUAGES = 981, ERR_NOLANGUAGE = 982, ERR_TEXTTOOSHORT = 983, ERR_NUMERIC_ERR = 999
};

}
#endif // IRCCODES_H
```

## 8.91 C++/customIrcServer/mainapp.cpp File Reference

`#include "mainapp.h"`

Include dependency graph for mainapp.cpp:



### Namespaces

- namespace server

## 8.92 mainapp.cpp

```
#include "mainapp.h"

namespace server {

mainApp::mainApp(QString parentPath, QObject *parent)
    :QObject(parent)
    ,users(new ircUserList())
    ,log(clsLog(LOGTAGS_MAIN))
{
    this->settings = new clsSettings(parentPath);
}


void mainApp::run(){
    this->log << "Starting up" << endl;

    this->irc = new ircClient(this->settings->getServer(),
                              this->settings->getChannel(),
                              this->settings->getUserName());
    this->ui = new uiServer(this->settings->getListeningPort(), this);

    connect(this->irc,SIGNAL(chatReceived(QString,nickAndStatus,QString)), this,
            SLOT(chatReceived(QString,nickAndStatus,QString)));

    connect(this->irc, SIGNAL(userOnline(nickAndStatus, QString, QString)), this,
            SLOT(userOnline(nickAndStatus, QString, QString )));

    connect(this->irc, SIGNAL(userOffline(nickAndStatus, QString, QString)), this,
            SLOT(userOffline(nickAndStatus, QString, QString )));


```

```
    connect(this->irc, SIGNAL(userChangeNick(nickAndStatus, nickAndStatus,
                                                QString))
            ,this,SLOT(userChangeNick(nickAndStatus, nickAndStatus,QString)));

    connect(this->irc, SIGNAL(userChangeStatus(nickAndStatus, nickAndStatus,
                                                  QString))
            ,this, SLOT(userChangeStatus(nickAndStatus, nickAndStatus,QString)));

    connect(this->irc, SIGNAL(IRCConnectTimeOut()), this, SLOT(ircConnectTimeOut()));
    connect(this->irc, SIGNAL(IRCConnected()), this, SLOT(ircConnected()));
    connect(this->irc, SIGNAL(IRCDisconnected(QString)), this,
            SLOT(ircDisconnected(QString)));

    connect(this->ui,SIGNAL(sgnChatReceived(QString)),this,SLOT(uiSendChat(QString)));
    connect(this->ui,SIGNAL(sgnUserQuery(uiClient*)), this,SLOT(doUserQuery(uiClient*)));
    connect(this->ui, SIGNAL(sgnSendEvents(uiClient*)), this, SLOT(doSendEvents(uiClient*)));
    connect(this->ui,SIGNAL(sgnChangeOwnUserStatus(QString)),
            this, SLOT(ownUserChangeStatus(QString)));


    //Init db of 'standard' irc users:
    ownUser = this->settings->getOwnUser();
    ownUser->setOnline(true);
    ownUser->setStatus("");
    this->irc->setOwnUser(ownUser);
    this->users->add(ownUser);
    foreach(ircUser* user, this->settings->getUsers()){
        this->users->add(user);
    }

    this->serverMessage("Connecting to irc.", SERVERMESSAGE_LEVEL_INFO);
    this->irc->doConnect();
    this->ui->startListen();
}

void mainApp::addToEventHistory(clsEvent *event){
    this->events.append(event);
    while(this->events.count() > MAX_EVENTS){
        this->events.pop_front();
    }
}

void mainApp::chatReceived(const QString channel, const nickAndStatus user,
                           const QString message){
    Q_UNUSED(channel);
    this->log << "Chat received:\t<" << user.nick << ">\t" << message << endl;

    //Create event:
    eventChat *chatEvent = new eventChat(
                this->users->getUserByNick(user)->getId(), message);

    this->addToEventHistory(chatEvent);
    jsonCommand chatMessage(JSONCOMMAND_EVENT);
    chatMessage.addToData("event",chatEvent->toVariant());

    this->ui->send(chatMessage);

}


void mainApp::userOnline(const nickAndStatus nick, const QString ircId, const QString channel){
    Q_UNUSED(channel);
    this->log << "Online\t" << nick.nick << "\t" << nick.status << endl;
    ircUser *user = this->users->getUser(nick, ircId);
    if(user->getName() == "NaN"){
        user = new ircUser(nick,ircId);
        this->users->add(user);
    }else{
        user->setStatus(nick.status);
        user->setOnline(true);
    }
    //Create event
    eventUserJoin *joinEvent = new eventUserJoin(
                user->getId(),user->toVariantMap());
    this->addToEventHistory(joinEvent);
```

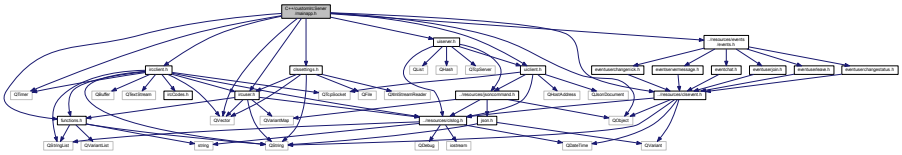
```
    jsonCommand uUser(JSONCOMMAND_EVENT);

    uUser.addToData("event",joinEvent->toVariant());
    this->ui->send(uUser);
}

void mainApp::userOffline(const nickAndStatus nick, const QString id, const QString channel){
    Q_UNUSED(channel);
    this->log << "Offline\t" << nick.nick << "\t" << nick.status << endl;
    ircUser *user = this->users->getUser(nick, id);
    if(user->getName() == "NaN"){
        //Say WHAT!?
        this->log << nick.nick << " bestaat niet!?" << endl;
    }else{
        user->setStatus("");
        user->setOnline(false);
        if(user->isStandard() == false){
            this->users->del(user);
        }

        //Create event
        eventUserLeave *leaveEvent = new eventUserLeave(
                    user->getId());
        this->addToEventHistory(leaveEvent);
        jsonCommand uUser(JSONCOMMAND_EVENT);

        uUser.addToData("event",leaveEvent->toVariant());
        this->ui->send(uUser);
    }
}

void mainApp::userChangeNick(const nickAndStatus oldNick,
                             const nickAndStatus newNick, const QString id){
    this->log << oldNick.nick << " changed nick to " << newNick.nick << endl;

    ircUser *user = this->users->getUser(oldNick, id);
    if(user->getName() == "NaN"){
        //Say WHAT!?
        this->log << oldNick.nick << " bestaat niet!?" << endl;
    }else{
        user->setNick(newNick.nick);

        //Create event
        eventUserChangeNick *changeEvent = new eventUserChangeNick(
                    user->getId(),
                    newNick.nick,
                    oldNick.nick);
        this->addToEventHistory(changeEvent);
        jsonCommand uUser(JSONCOMMAND_EVENT);

        uUser.addToData("event",changeEvent->toVariant());
        this->ui->send(uUser);
    }
}

void mainApp::userChangeStatus(const nickAndStatus oldNick,
                               const nickAndStatus newNick, const QString id){
    this->log << oldNick.nick << " changed status to " << newNick.status << endl;

    ircUser *user = this->users->getUser(oldNick,id);
    if(user->getName() == "NaN"){
        //Say WHAT!?
        this->log << oldNick.nick << " bestaat niet!?" << endl;
    }else{
        user->setStatus(newNick.status);
        //Create event
        eventUserChangeStatus *changeEvent = new eventUserChangeStatus(
                    user->getId(),
                    newNick.status,
                    oldNick.status);
        this->addToEventHistory(changeEvent);
        jsonCommand uUser(JSONCOMMAND_EVENT);

        uUser.addToData("event",changeEvent->toVariant());
        this->ui->send(uUser);
    }

}

void mainApp::uiSendChat(const QString message){
    this->irc->sendChat(message);
```

```
}

void mainApp::doUserQuery(uiClient *client){
    //Send all the users back
    jsonCommand uQuery(JSONCOMMAND_USERQUERY);
    QList<QVariant> users;
    foreach(ircUser *user, this->users->getAll()){
        users.append(QVariant(user->toVariantMap()));
    }
    uQuery.addToData("users",QVariant::fromValue(users));

    this->ui->send(client,uQuery);

}

void mainApp::doSendEvents(uiClient *client){
    jsonCommand toSend(JSONCOMMAND_GETEVENTS);
    QList<QVariant> events;
    foreach(clsEvent *ev, this->events){
        events.append(ev->toVariant());
    }
    toSend.addToData("events",QVariant(events));

    this->ui->send(client, toSend);
}

void mainApp::ownUserChangeStatus(QString newStatus){
    this->ownUser->setStatus(newStatus);
    if(newStatus == "Offline"){
        this->ownUser->setOnline(false);
    }else{
        this->ownUser->setOnline(true);
    }
    if(this->ownUser->getStatus() == ""){
        this->irc->setOwnNick(this->ownUser->getName());
    }else{
        this->irc->setOwnNick(this->ownUser->getName() + "|" +
                              this->ownUser->getStatus());
    }
}

void mainApp::serverMessage(const QString &message, const level lvl){
    eventServerMessage *myMessage = new
            eventServerMessage(message, lvl);

    jsonCommand eventMsg(JSONCOMMAND_EVENT);

    eventMsg.addToData("event",myMessage->toVariant());
    this->ui->send(eventMsg);
    this->addToEventHistory(myMessage);
}

void mainApp::ircConnectTimeOut(){
    this->serverMessage(
                QString("Connection to IRC failed. Reconnecting in %1 seconds").
                        arg(qint32(RECONNECT_IN_S)), SERVERMESSAGE_LEVEL_ERROR);

    //Now, let's reconnect in RECONNECT_IN_S
    this->reconnectTimer.singleShot(RECONNECT_IN_S*1000, this,
                                    SLOT(doIrcReconnect()));
}

void mainApp::ircConnected(){
    this->serverMessage("Connected to IRC!", SERVERMESSAGE_LEVEL_INFO);
}

void mainApp::ircDisconnected(QString reason){
    QString msg("Disconnected from IRC: %1. Reconnecting in %2 seconds");
    msg = msg.arg(reason, qint32(RECONNECT_IN_S));

    this->serverMessage(msg, SERVERMESSAGE_LEVEL_ERROR);
    //Reconnect in RECONNECT_IN_S
    this->reconnectTimer.singleShot(RECONNECT_IN_S*1000, this,
                                    SLOT(doIrcReconnect()));
}

void mainApp::doIrcReconnect(){
    this->serverMessage("Reconnecting", SERVERMESSAGE_LEVEL_INFO);
    this->irc->doConnect();
}

}
```

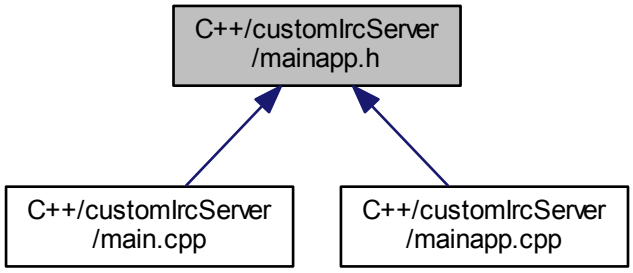
## 8.93 C++/customIrcServer/mainapp.h File Reference

```
#include <QObject>
#include <QVector>
#include <QTimer>
#include "ircclient.h"
#include "functions.h"
#include "ircuser.h"
#include "uiserver.h"
#include "uiclient.h"
#include "clssettings.h"
#include "../resources/clsevent.h"
#include "../resources/events/events.h"
```

Include dependency graph for mainapp.h:



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




### Classes

- class server::mainApp

  *The main application.*

### Namespaces

- namespace server

### Macros

- #define MAX_EVENTS 5000
- #define RECONNECT_IN_S 10

### 8.93.1 Macro Definition Documentation

#### 8.93.1.1 #define MAX_EVENTS 5000

The maximum number of events stored before deleting the earliest

Definition at line 16 of file mainapp.h.

#### 8.93.1.2 #define RECONNECT_IN_S 10

Ammount of seconds it should take before starting to reconnect to the IRC

Definition at line 17 of file mainapp.h.

## 8.94 mainapp.h

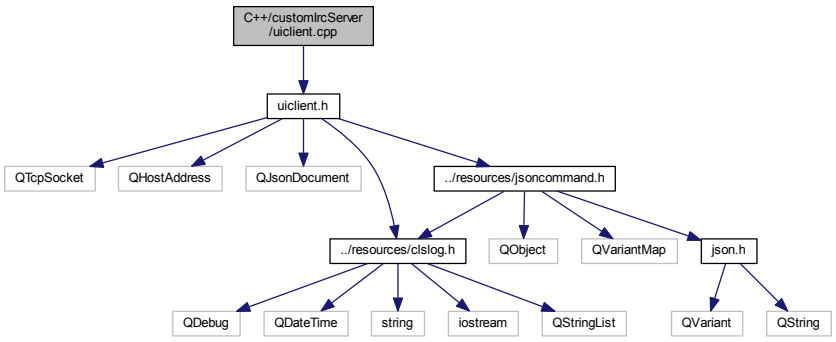
```
00001 #ifndef MAINAPP_H
00002 #define MAINAPP_H
00003
00004 #include <QObject>
00005 #include <QVector>
00006 #include <QTimer>
00007 #include "ircclient.h"
00008 #include "functions.h"
00009 #include "ircuser.h"
00010 #include "uiserver.h"
00011 #include "uiclient.h"
00012 #include "clssettings.h"
00013 #include "../resources/clsevent.h"
00014 #include "../resources/events/events.h"
00015
00016 #define MAX_EVENTS  5000
00017 #define RECONNECT_IN_S  10
00018 using namespace shared;
00019 using namespace shared::events;
00020 namespace server {
00025 class mainApp : public QObject
00026 {
00027     Q_OBJECT
00028 public:
00029     explicit mainApp(QString parentPath, QObject *parent = 0);
00030     void run();
00031
00036     void serverMessage(const QString &message, const level lvl);
00037 private:
00043     void addToEventHistory(clsEvent *event);
00044
00045 signals:
00046
00047 public slots:
00054     void chatReceived(const QString channel, const nickAndStatus user,
00055                       const QString message);
00056
00063     void userOnline(const nickAndStatus nick, const QString ircId,
00064                     const QString channel);
00065
00072     void userOffline(const nickAndStatus nick, const QString id,
00073                      const QString channel);
00074
00082     void userChangeNick(const nickAndStatus oldNick,
00083                         const nickAndStatus newNick,
00084                         const QString id);
00085
00093     void userChangeStatus(const nickAndStatus oldNick,
00094                           const nickAndStatus newNick,
00095                           const QString id);
00096
00101     void uiSendChat(const QString message);
00102
00107     void doUserQuery(uiClient *client);
00112     void doSendEvents(uiClient *client);
00113
00118     void ownUserChangeStatus(QString newStatus);
00119
00123     void ircConnectTimeOut();
00124
00128     void ircConnected();
```

```
    void ircDisconnected(QString reason);

    void doIrcReconnect();
private:
    const static logTags TAG = LOGTAGS_MAIN;
    ircClient *irc;
    ircUserList *users;
    clsLog log;
    uiServer *ui;

    ircUser *ownUser;
    QVector<clsEvent *> events;

    clsSettings *settings;

    QTimer reconnectTimer;

};
}
#endif // MAINAPP_H
```

## 8.95 C++/customIrcServer/uiclient.cpp File Reference

`#include "uiclient.h"`

Include dependency graph for uiclient.cpp:




### Namespaces

- namespace server

## 8.96 uiclient.cpp

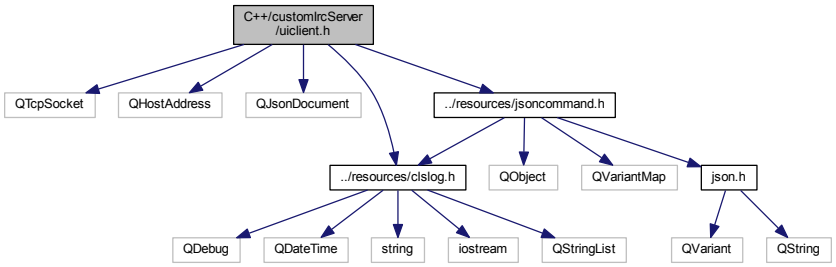
```
#include "uiclient.h"

namespace server {

uiClient::uiClient(QTcpSocket *parent)
    :sock(parent)
    ,log(LOGTAGS_UI)
    ,depth(0)
{

    QHostAddress myRemote = this->sock->peerAddress();
    this->log.setPrefix(myRemote.toString());
    this->log << "Initialized tcp client " << myRemote.toString() << endl;
    this->buffer = new QString();

    connect(this->sock, SIGNAL(disconnected()), SLOT(
removeConnection()));
    connect(this->sock, SIGNAL(readyRead()),    SLOT(receiveMessage()));
}
```

```
00019
00020
00021 void uiClient::send(QString data){
00022     this->sock->write(data.toStdString().c_str());
00023 }
00024
00025 void uiClient::removeConnection(){
00026     this->log << "Connection lost" << endl;
00027     //Tell our parent class to remove this item
00028     emit this->disconnected(this);
00029 }
00030
00031
00032 bool uiClient::checkForJsonCommand(QByteArray toAdd,
      jsonCommand &comm){
00033     bool ret = false;
00034     foreach(char b, toAdd){
00035         if(b == '{' || b == '['){
00036             depth++;
00037         }else if(b == '}' || b == ']'){
00038             depth--;
00039         }
00040         this->buffer->append(b);
00041
00042         if(depth == 0){
00043             //Clear the buffer.
00044             if(comm.getCommand() == JSONCOMMAND_NONE){
00045                 ret = true;
00046                 comm = new jsonCommand(*this->buffer);
00047                 this->log << "Data received: \t" << this->buffer <<
      endl;
00048                 this->buffer->clear();
00049             }
00050         }
00051     }
00052     return ret;
00053 }
00054
00055 void uiClient::receiveMessage(){
00056     jsonCommand comm (JSONCOMMAND_NONE);
00057     if(this->checkForJsonCommand(this->sock->readAll(), comm)){
00058         switch (comm.getCommand()) {
00059             case JSONCOMMAND_OWNCHAT:
00060                 emit this->chatReceived(comm.getData("chat").toString());
00061                 break;
00062             case JSONCOMMAND_USERQUERY:
00063                 emit this->userQuery(this);
00064                 break;
00065             case JSONCOMMAND_CHANGEOWNUSER:
00066                 emit this->changeOwnUser(comm);
00067                 break;
00068             case JSONCOMMAND_GETEVENTS:
00069                 emit this->getEvents(this);
00070                 break;
00071             default:
00072                 break;
00073         }
00074     }
00075 }
00076
00077 }
```

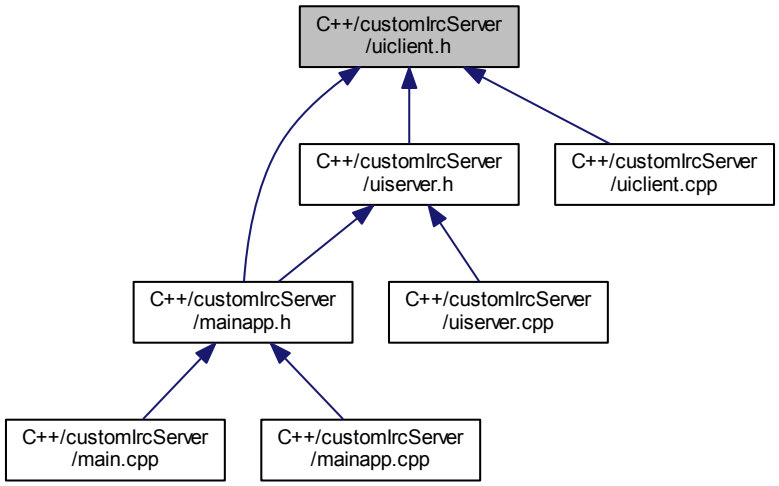
## 8.97 C++/customIrcServer/uiclient.h File Reference

```
#include <QTcpSocket>
#include <QHostAddress>
#include <QJsonDocument>
#include "../resources/clslog.h"
#include "../resources/jsoncommand.h"
```

Include dependency graph for uiclient.h:




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




### Classes

- class server::uiClient

  *The uiClient class.*

### Namespaces

- namespace server

## 8.98 uiclient.h

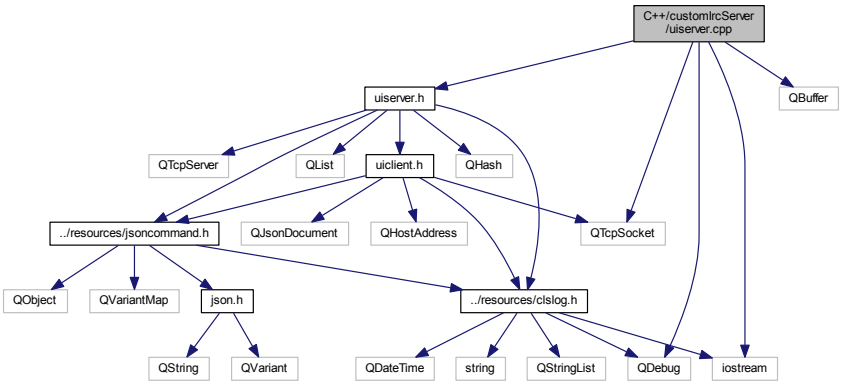
```
00001 #ifndef UICLIENT_H
00002 #define UICLIENT_H
00003
00004 #include <QTcpSocket>
00005 #include <QHostAddress>
00006 #include <QJsonDocument>
```

```
00007
00008 #include "../resources/clslog.h"
00009 #include "../resources/jsoncommand.h"
00010
00011 using namespace K4U;
00012 namespace server {
00017 class uiClient: public QObject{
00018     Q_OBJECT
00019 public:
00024     uiClient(QTcpSocket *parent);
00025
00030     void send(QString data);
00031
00032
00033 private:
00045     bool checkForJsonCommand(QByteArray toAdd, jsonCommand &comm);
00046 signals:
00047
00052     void disconnected(uiClient *client);
00057     void chatReceived(QString message);
00062     void userQuery(uiClient *client);
00067     void getEvents(uiClient *client);
00068
00074     void changeOwnUser(jsonCommand &comm);
00075
00076 public slots:
00077     void removeConnection();
00078     void receiveMessage();
00079
00080 private:
00081     QTcpSocket *sock;
00082     clsLog log;
00083     QString *buffer;
00084     int depth;
00085 };
00086
00087 }
00088 #endif // UICLIENT_H
```

## 8.99 C++/customIrcServer/uiserver.cpp File Reference

```
#include "uiserver.h"
#include <QBuffer>
#include <QTcpSocket>
#include <iostream>
#include <QDebug>
```

Include dependency graph for uiserver.cpp:




### Namespaces

- namespace server

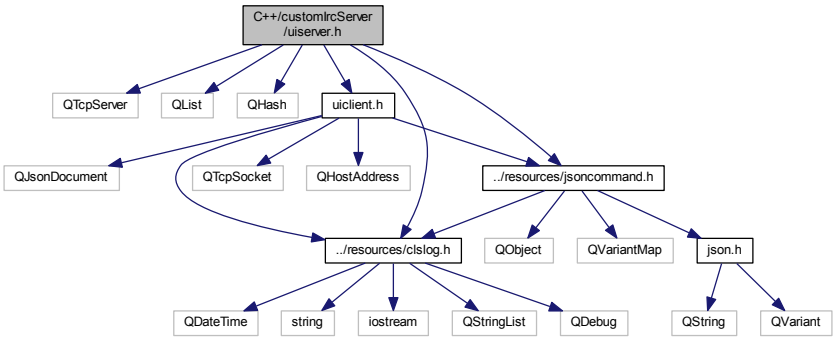
## 8.100 uiserver.cpp

```
#include "uiserver.h"
#include <QBuffer>
#include <QTcpSocket>
#include <iostream>
#include <QDebug>

namespace server {
uiServer::uiServer(int port, QObject* parent)
    :QTcpServer(parent)
    ,log(LOGTAGS_SERVER)
    ,port(port)
{
    connect(this, SIGNAL(newConnection()), this, SLOT(acceptConnection()));
}

uiServer::~uiServer(){ }

void uiServer::startListen(){
    this->log << "Started listening on port " << this->port << endl;
    this->listen(QHostAddress::Any,this->port);
}


void uiServer::send(const jsonCommand &toSend){
    //this->log << "Sending json object: " << docToSend->toJson() << endl;
    this->log << "Sending JSON: " << toSend.toJsonString() << endl;

    foreach(uiClient *cl, this->connections){
        cl->send(toSend.toJsonString() + "\r\n");
    }
}

void uiServer::send(uiClient *client, const jsonCommand &toSend){
    //this->log << "Sending json object: " << docToSend->toJson() << endl;
    this->log << "Sending JSON: " << toSend.toJsonString() << endl;

    client->send(toSend.toJsonString() + "\r\n");
}

void uiServer::acceptConnection(){
    uiClient *connection = new uiClient(this->nextPendingConnection());
    connect(connection, SIGNAL(disconnected(uiClient*)),
            this, SLOT(lostConnection(uiClient*)));
    connect(connection, SIGNAL(chatReceived(QString)),this,SLOT(chatReceived(QString)));
    connect(connection, SIGNAL(userQuery(uiClient*)), this, SLOT(doUserQuery(uiClient*)));
    connect(connection, SIGNAL(changeOwnUser(jsonCommand&)),
            this, SLOT(changeOwnUser(jsonCommand&)));
    connect(connection, SIGNAL(getEvents(uiClient*)), this, SLOT(doSendEvents(uiClient*)));
    connections.append(connection);
}

void uiServer::lostConnection(uiClient *client){
    this->connections.removeAll(client);
}

void uiServer::chatReceived(QString message){
    emit this->sgnChatReceived(message);
}

void uiServer::doUserQuery(uiClient *client){
    emit this->sgnUserQuery(client);
}

void uiServer::doSendEvents(uiClient *client){
    emit this->sgnSendEvents(client);
}

void uiServer::changeOwnUser(jsonCommand &comm){
    if(comm.getData("change") == "STATUS"){
        emit this->sgnChangeOwnUserStatus(comm.getData("new").toString());
    }
    //Immediately forward this to other clients:
    this->send(comm);
}

}
```

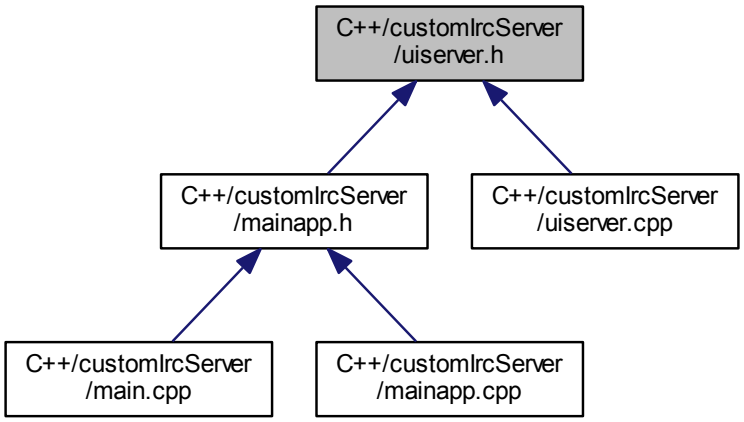
## 8.101 C++/customIrcServer/uiserver.h File Reference

```
#include <QTcpServer>
#include <QList>
#include <QHash>
#include "uiclient.h"
#include "../resources/clslog.h"
#include "../resources/jsoncommand.h"
```

Include dependency graph for uiserver.h:




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




### Classes

- class server::uiServer

  *The uiServer class.*

### Namespaces

- namespace server

### Macros

- #define DEFAULT_LISTENING_PORT 1337

### 8.101.1 Macro Definition Documentation

#### 8.101.1.1 #define DEFAULT_LISTENING_PORT 1337

The default listening port for the UI server

Definition at line 11 of file uiserver.h.

## 8.102 uiserver.h

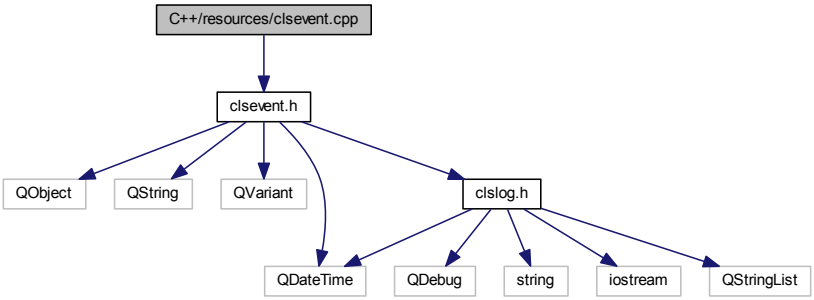
```
00001 #ifndef SIMPLECHATSERVER_H
00002 #define SIMPLECHATSERVER_H
00003
00004 #include <QTcpServer>
00005 #include <QList>
00006 #include <QHash>
00007 #include "uiclient.h"
00008 #include "../resources/clslog.h"
00009 #include "../resources/jsoncommand.h"
00010
00011 #define DEFAULT_LISTENING_PORT 1337
00013 class QTcpSocket;
00014
00015 namespace server {
00021 class uiServer : public QTcpServer
00022 {
00023     Q_OBJECT
00024
00025 public:
00031     uiServer(int port = DEFAULT_LISTENING_PORT, QObject* parent = 0);
00032     ~uiServer();
00033
00037     void startListen();
00038
00039
00044     void send(const jsonCommand &toSend);
00045
00051     void send(uiClient *client, const jsonCommand &toSend);
00052
00053 private slots:
00057     void acceptConnection();
00058
00063     void lostConnection(uiClient* client);
00064
00069     void chatReceived(QString message);
00070
00075     void doUserQuery(uiClient *client);
00076
00081     void doSendEvents(uiClient *client);
00082
00088     void changeOwnUser(jsonCommand &comm);
00089 signals:
00094     void sgnChatReceived(QString message);
00095
00101     void sgnUserQuery(uiClient *client);
00107     void sgnSendEvents(uiClient *client);
00108
00113     void sgnChangeOwnUserStatus(QString newStatus);
00114
00115
00116 private:
00117     QList<uiClient*> connections;
00118
00119     clsLog log;
00120
00121     int port;
```

```
};

}
#endif // SIMPLECHATSERVER_H
```

## 8.103 C++/resources/clsevent.cpp File Reference

`#include "clsevent.h"`

Include dependency graph for clsevent.cpp:




### Namespaces

- namespace shared
- namespace shared::events

## 8.104 clsevent.cpp

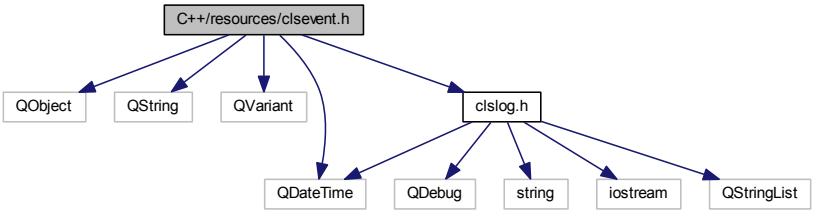
```
#include "clsevent.h"

namespace shared{
    namespace events{
        clsEvent::clsEvent()
            :log(LOGTAGS_EVENT)
        {
            this->timeOfEvent = QDateTime::currentDateTime();
        }

        clsEvent::clsEvent(QVariantMap data)
            :log(LOGTAGS_EVENT)
        {
            this->data = data["data"].toMap();//Makes sense?
            this->setEventType(data["type"].toString());
            this->timeOfEvent = data["dateTime"].toDateTime();
        }

        void clsEvent::addToData(const QString &key, const QVariant data){
            this->data.insert(key, data);
        }

        QVariant clsEvent::toVariant() const{
            QVariantMap ret;
            ret["type"] = this->getEventType();
            ret["dateTime"] = this->timeOfEvent;
            ret["data"] = QVariant(this->data);

            return QVariant(ret);
        }

        QVariant clsEvent::getData(QString key) const{
            return this->data[key];
        }

```

```
        eventTypes clsEvent::getType() const{
            return this->ourType;
        }


        QTime clsEvent::getTime() const{
            return this->timeOfEvent.time();
        }

        void clsEvent::setEventType(const QString eventString){
            eventTypes ret = EVENTTYPE_NONE;
            if(eventString == "CHAT"){
                ret = EVENTTYPE_CHAT;
            }else if(eventString == "USER_JOIN"){
                ret = EVENTTYPE_USER_JOIN;
            }else if(eventString == "USER_LEFT"){
                ret = EVENTTYPE_USER_LEFT;
            }else if(eventString == "USER_STATUS"){
                ret = EVENTTYPE_USER_STATUS;
            }else if(eventString == "USER_NICK"){
                ret = EVENTTYPE_USER_NICK;
            }else if(eventString == "SERVERMSG"){
                ret = EVENTTYPE_SERVERMSG;
            }
            this->ourType = ret;
        }

        QString clsEvent::getEventType() const{
            QString ret = "";
            switch(this->ourType){
                case EVENTTYPE_CHAT:
                    ret = "CHAT";
                    break;
                case EVENTTYPE_USER_JOIN:
                    ret = "USER_JOIN";
                    break;
                case EVENTTYPE_USER_LEFT:
                    ret = "USER_LEFT";
                    break;
                case EVENTTYPE_USER_STATUS:
                    ret = "USER_STATUS";
                    break;
                case EVENTTYPE_USER_NICK:
                    ret = "USER_NICK";
                    break;
                case EVENTTYPE_SERVERMSG:
                    ret = "SERVERMSG";
                    break;
                case EVENTTYPE_NONE:
                    ret = "NaN";
                    break;
            }
            return ret;
        }


    }
}
```

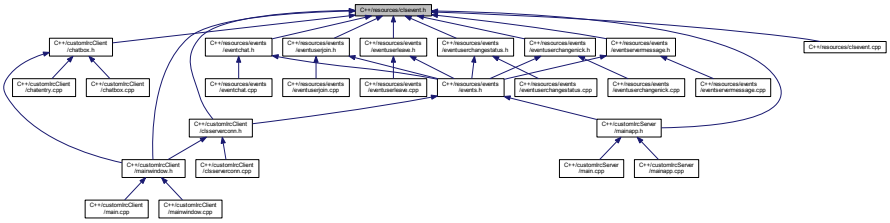
## 8.105 C++/resources/clsevent.h File Reference

```
#include <QObject>
#include <QString>
#include <QVariant>
#include <QDateTime>
#include "clslog.h"
```

Include dependency graph for clsevent.h:




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



## Classes

- class shared::events::clsEvent

  *The main event class.*

## Namespaces

- namespace shared
- namespace shared::events

## Enumerations

- enum shared::events::eventTypes { shared::events::EVENTTYPE_NONE = 0x00, shared::events::EVENTTYPE_CHAT, shared::events::EVENTTYPE_USER_LEFT, shared::events::EVENTTYPE_USER_STATUS, shared::events::EVENTTYPE_USER_JOIN, shared::events::EVENTTYPE_USER_NICK, shared::events::EVENTTYPE_SERVERMSG }

  *The eventTypes enum.*

# 8.106 clsevent.h

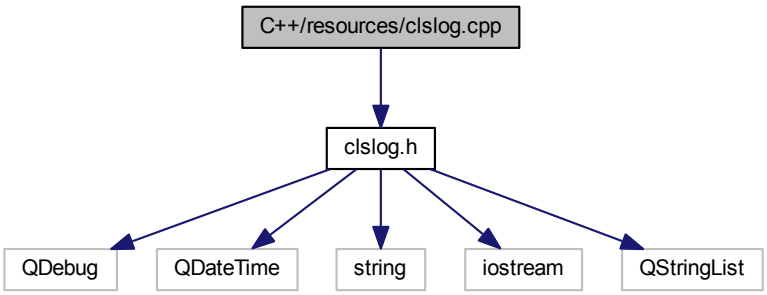
```
00001 #ifndef CLSEVENT_H
00002 #define CLSEVENT_H
00003
00004 #include <QObject>
00005 #include <QString>
00006 #include <QVariant>
00007 #include <QDateTime>
00008 #include "clslog.h"
```

```
00009
00010 using namespace K4U;
00011 namespace shared{
00012
00013     namespace events{
00017         enum eventTypes{
00018             EVENTTYPE_NONE = 0x00,
00019             EVENTTYPE_CHAT,
00020             EVENTTYPE_USER_LEFT,
00021             EVENTTYPE_USER_STATUS,
00022             EVENTTYPE_USER_JOIN,
00023             EVENTTYPE_USER_NICK,
00024             EVENTTYPE_SERVERMSG
00025         };
00026
00027
00032         class clsEvent
00033         {
00034             public:
00038                 explicit clsEvent();
00044                 explicit clsEvent(QVariantMap data);
00045
00051                 void addToData(const QString &key, const QVariant data);
00052
00057                 QVariant toVariant() const;
00058
00064                 QVariant getData(QString key) const;
00065
00070                 eventTypes getType() const;
00071
00076                 QTime getTime() const;
00077
00078             protected:
00082                 QVariantMap data;
00083                 eventTypes ourType;
00084                 QDateTime timeOfEvent;
00085
00090                 QString getEventType() const;
00091
00096                 void setEventType(const QString eventString);
00097
00098                 clsLog log;
00099         };
00100     }
00101 }
00102 #endif // CLSEVENT_H
```

## 8.107 C++/resources/clslog.cpp File Reference

`#include "clslog.h"`

Include dependency graph for clslog.cpp:




### Namespaces

- namespace K4U

*Contains functions used for more projects.*

### Functions

- clsLog & K4U::endl (clsLog &log)

## 8.108 clslog.cpp

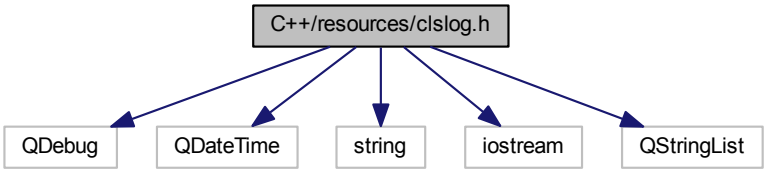
```
#include "clslog.h"

using namespace std;

namespace K4U{

clsLog::clsLog(logTags tag)
    :isFirst(true)
    ,tag(tag)
{}


clsLog &clsLog::operator <<(clsLog &(*manip)(clsLog &)){
    return manip(*this);
}

clsLog &clsLog::operator <<(bool msg){
    if(msg == true){
        std::cout << "True";
    }else{
        std::cout << "False";
    }
    return *this;
}

clsLog &clsLog::operator <<(QVariantMap &map){
    foreach(QString key, map.keys()){
        *this << key << "\t"
                  << map[key] << endl;
    }
    return *this;
}

clsLog &clsLog::operator <<(const QVariantMap &map){
    foreach(const QString key, map.keys()){
        *this << key << "\t"
                  << map[key] << endl;
    }
    return *this;
}

clsLog &clsLog::operator <<(const QVariant &msg){
    *this << msg.toString();

    return *this;
}

clsLog &clsLog::operator <<(QVariant &msg){
    *this << msg.toString();

    return *this;
}

clsLog &clsLog::operator <<(QByteArray &bArr){
    *this << QString(bArr);
    return *this;
}

clsLog &clsLog::operator <<(const QByteArray &bArr){
    *this << QString(bArr);
    return *this;
}


clsLog &endl(clsLog &log){
    log.doEndl();
    return log;
}

void clsLog::doEndl(){
    std::cout << std::endl;
```

```
    isFirst = true;
}

void clsLog::setPrefix(QString prefix){
    this->prf = prefix;
}

void clsLog::prefix(){
    if(isFirst == true){
        isFirst = false;
        QDateTime dateTime = QDateTime::currentDateTime();
        QString dateTimeString = dateTime.toString();

        std::cout << "[" << this->getTag() << "]\t";
        std::cout << dateTimeString.toStdString() << "\t";
        if(this->prf != ""){
            std::cout << this->prf.toStdString() << "\t";
        }else{
            std::cout << "\t";
        }
        std::cout << "\t";
    }
}

clsLog& clsLog::operator <<(QString msg){
    this->prefix();
    std::cout << msg.toStdString();

    return *this;
}

clsLog &clsLog::operator <<(const QString *msg){
    this->prefix();
    std::cout << msg->toStdString();

    return *this;
}

clsLog& clsLog::operator <<(const char *msg){
    this->prefix();
    std::cout << msg;

    return *this;
}

clsLog& clsLog::operator <<(int msg){
    this->prefix();
    std::cout << msg;

    return *this;
}


std::string clsLog::getTag(){
    string ret = "";
    switch(this->tag){
        case LOGTAGS_IRC:
            ret = "IRC";
            break;
        case LOGTAGS_MAIN:
            ret = "MAIN";
            break;
        case LOGTAGS_USER:
            ret= "USER";
            break;
        case LOGTAGS_SERVER:
            ret = "SRVR";
            break;
        case LOGTAGS_UI:
            ret = "UI";
            break;
        case LOGTAGS_WEB:
            ret = "WEB";
            break;
        case LOGTAGS_EVENT:
            ret = "EVNT";
            break;
        default:
            ret = "UNK";
            break;
    }
    return ret;
}


}
```

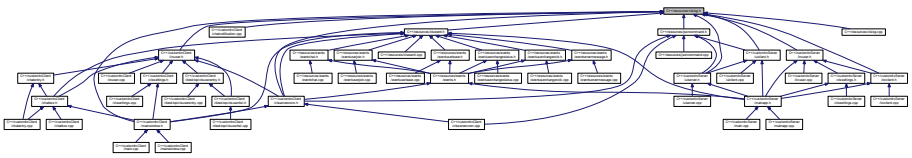
## 8.109 C++/resources/clslog.h File Reference

```
#include <QDebug>
#include <QDateTime>
#include <string>
#include <iostream>
#include <QStringList>
```

Include dependency graph for clslog.h:




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



### Classes

- class K4U::clsLog

  *Custom logging class.*

### Namespaces

- namespace K4U

  *Contains functions used for more projects.*

### Enumerations

- enum K4U::logTags {
  K4U::LOGTAGS_UNKNOWN = 0xFF, K4U::LOGTAGS_IRC = 0x00, K4U::LOGTAGS_MAIN, K4U::LOGTAGS_USER,
  K4U::LOGTAGS_SERVER, K4U::LOGTAGS_UI, K4U::LOGTAGS_WEB, K4U::LOGTAGS_EVENT }

### Functions

- clsLog & K4U::endl (clsLog &log)

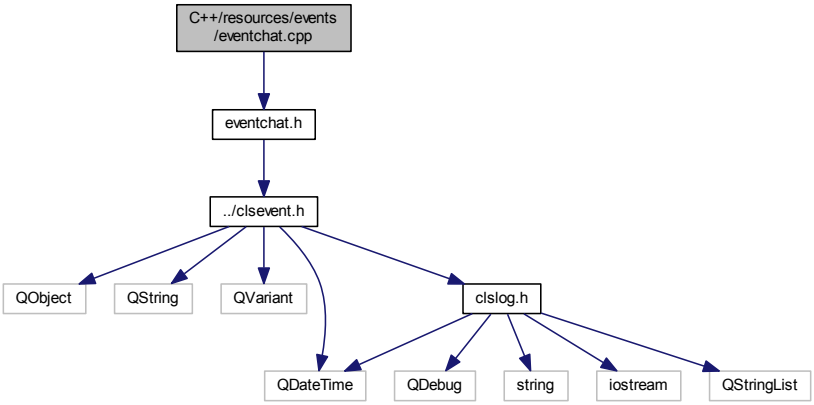
## 8.110 clslog.h

```
#ifndef CLSLOG_H
#define CLSLOG_H
#include <QDebug>
#include <QDateTime>
#include <string>
#include <iostream>
#include <QStringList>

namespace K4U {
enum logTags{
    LOGTAGS_UNKNOWN = 0xFF,
    LOGTAGS_IRC = 0x00,
    LOGTAGS_MAIN,
    LOGTAGS_USER,
    LOGTAGS_SERVER,
    LOGTAGS_UI,
    LOGTAGS_WEB,
    LOGTAGS_EVENT
};

class clsLog{
public:
    clsLog(logTags tag);

    clsLog& operator<<(clsLog& (*manip)(clsLog&));

    clsLog& operator << (QString msg);
    clsLog& operator << (const QString *msg);
    clsLog& operator << (const char* msg);
    clsLog& operator << (int msg);
    clsLog& operator << (clsLog *log);
    clsLog& operator << (bool msg);
    clsLog& operator << (QVariantMap &map);
    clsLog& operator << (const QVariantMap &map);
    clsLog& operator << (const QVariant &msg);
    clsLog& operator << (QVariant &msg);
    clsLog& operator << (QByteArray &bArr);
    clsLog& operator << (const QByteArray &bArr);


    void doEndl();
    void setPrefix(QString prefix);
private:
    bool isFirst;
    logTags tag;

    void prefix();
    std::string getTag();
    QString prf;
};

clsLog& endl(clsLog& log);


}

#endif // CLSLOG_H
```

## 8.111 C++/resources/events/eventchat.cpp File Reference

```
#include "eventchat.h"
```

Include dependency graph for eventchat.cpp:




### Namespaces

- namespace shared
- namespace shared::events

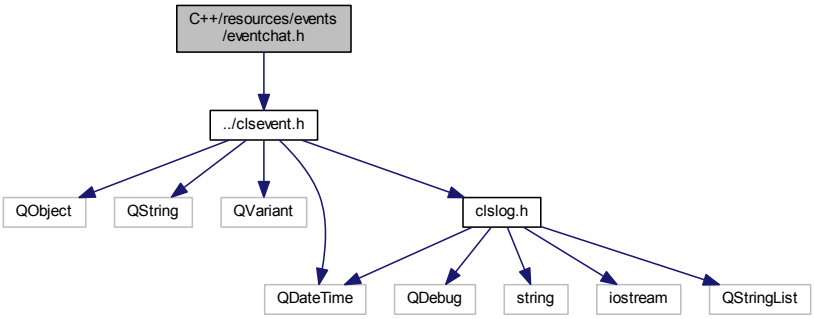
## 8.112 eventchat.cpp

```
#include "eventchat.h"

namespace shared{
    namespace events{

        eventChat::eventChat(int userId, QString message){
            ourType = EVENTTYPE_CHAT;
            this->addToData("userId",userId);
            this->addToData("message", message);
        }

        eventChat::eventChat(QVariantMap data)
            :clsEvent(data)
        { }

        int eventChat::getUser() const{
            return this->getData("userId").toInt();
        }

        QString eventChat::getMessage() const{
            return this->getData("message").toString();
        }


    }
}
```

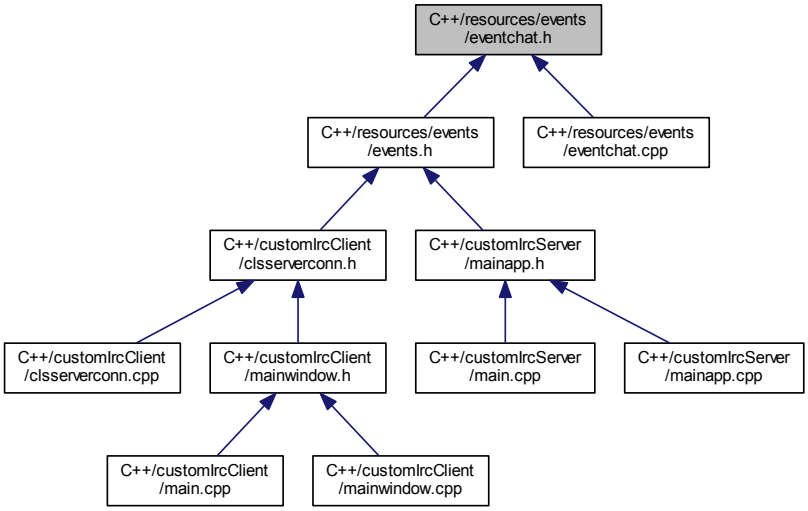
## 8.113 C++/resources/events/eventchat.h File Reference

```
#include "../clsevent.h"
```

Include dependency graph for eventchat.h:




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




### Classes

- class shared::events::eventChat

  *Class containing a chatEvent.*

### Namespaces

- namespace shared
- namespace shared::events

## 8.114 eventchat.h

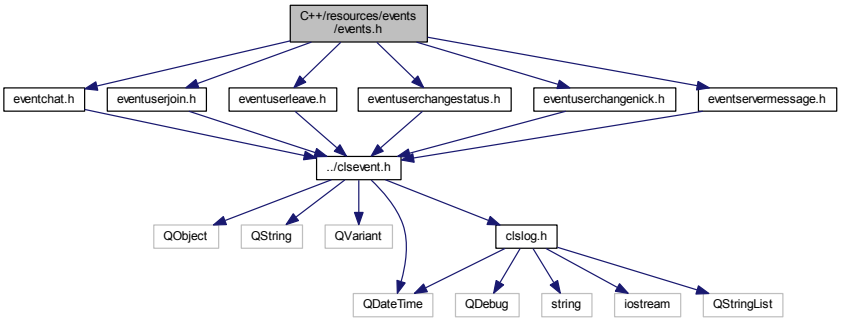
```
00001 #ifndef EVENTCHAT_H
```

```
#define EVENTCHAT_H
#include "../clsevent.h"

namespace shared{
    namespace events{
        class eventChat : public clsEvent
        {
            public:
                eventChat(int userId, QString message);

                eventChat(QVariantMap data);

                int getUser() const;
                QString getMessage() const;
        };
    }
}
#endif // EVENTCHAT_H
```

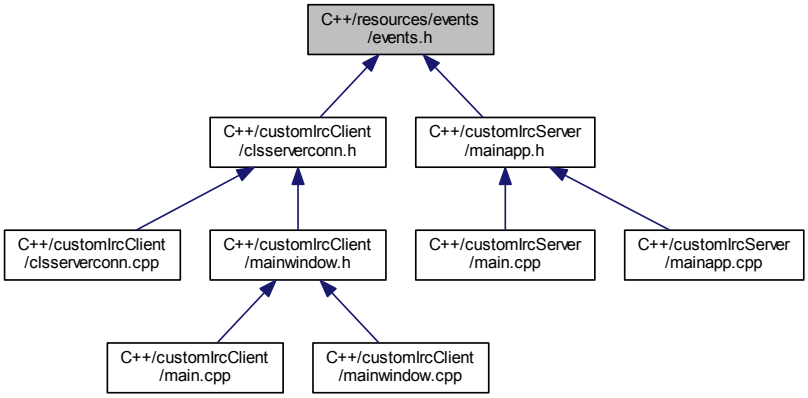
## 8.115 C++/resources/events/events.h File Reference

```
#include "eventchat.h"
#include "eventuserjoin.h"
#include "eventuserleave.h"
#include "eventuserchangestatus.h"
#include "eventuserchangenick.h"
#include "eventservermessage.h"
```

Include dependency graph for events.h:



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



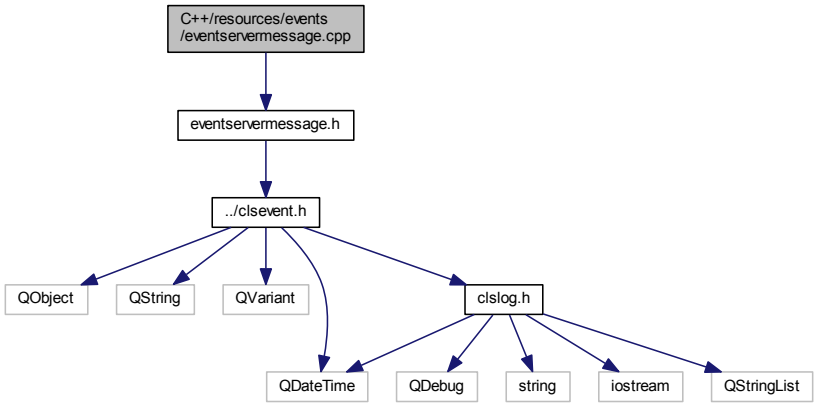

## 8.116 events.h

```
00001 #include "eventchat.h"
00002 #include "eventuserjoin.h"
00003 #include "eventuserleave.h"
00004 #include "eventuserchangestatus.h"
00005 #include "eventuserchangenick.h"
00006 #include "eventservermessage.h"
00007
00008 #ifndef EVENTS_H
00009 #define EVENTS_H
00010
00011 #endif // EVENTS_H
```

## 8.117 C++/resources/events/eventservermessage.cpp File Reference

`#include "eventservermessage.h"`

Include dependency graph for eventservermessage.cpp:



### Namespaces

- namespace shared
- namespace shared::events

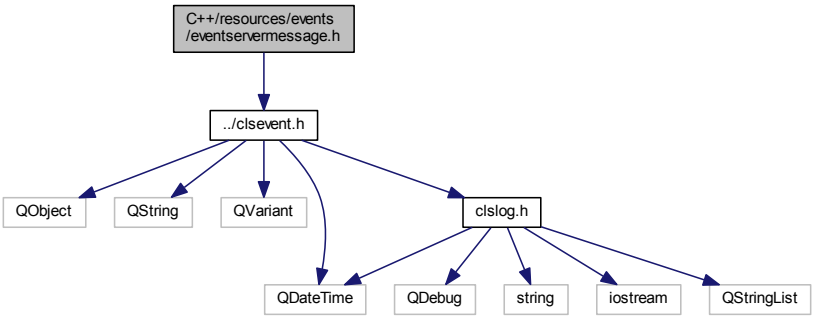
## 8.118 eventservermessage.cpp

```
#include "eventservermessage.h"

namespace shared{
    namespace events{

        eventServerMessage::eventServerMessage(QString message, level lvl){
            ourType = EVENTTYPE_SERVERMSG;
            this->addToData("message", message);
            this->addToData("level",lvl);
        }

        eventServerMessage::eventServerMessage(QVariantMap data)
            :clsEvent(data)
        { }

        QString eventServerMessage::getMessage() const{
            return this->getData("message").toString();
        }

        level eventServerMessage::getLevel() const {
            return (level)this->getData("level").toInt();
        }


    }
}
```

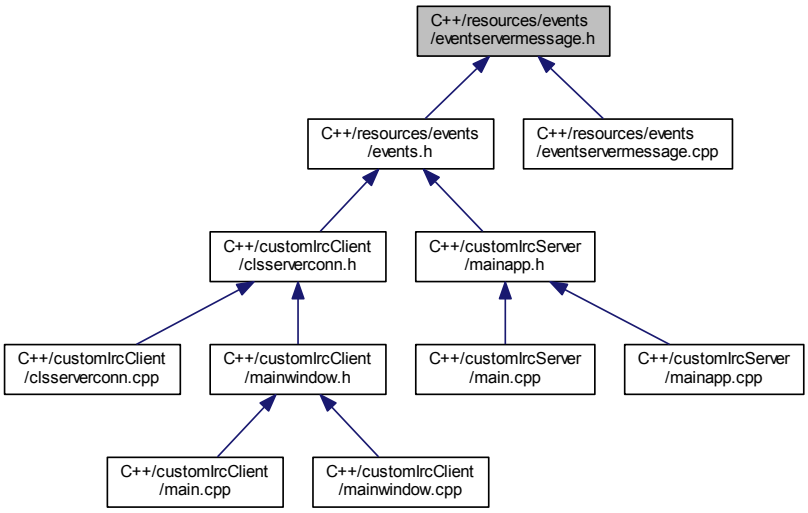
## 8.119 C++/resources/events/eventservermessage.h File Reference

`#include "../clsevent.h"`

Include dependency graph for eventservermessage.h:



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




### Classes

- class shared::events::eventServerMessage

  *Class that contains data for when the server wants to say something.*

### Namespaces

- namespace shared
- namespace shared::events

### Enumerations

- enum shared::level { shared::SERVERMESSAGE_LEVEL_ERROR = 0x01, shared::SERVERMESSAGE_LEVEL_INFO }

## 8.120 eventservermessage.h

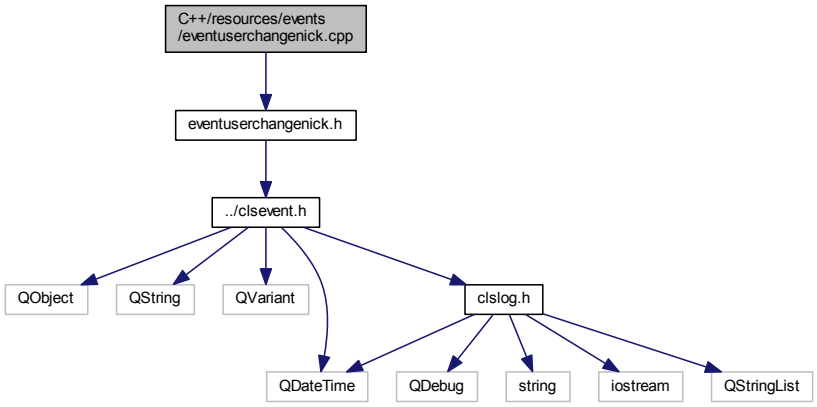
```
#ifndef EVENTSERVERMESSAGE_H
#define EVENTSERVERMESSAGE_H
#include "../clsevent.h"

namespace shared {
    enum level{
        SERVERMESSAGE_LEVEL_ERROR = 0x01,
        SERVERMESSAGE_LEVEL_INFO
    };
    namespace events {
        class eventServerMessage : public clsEvent
        {
        public:
            eventServerMessage(QString message, level lvl);

            eventServerMessage(QVariantMap data);

            QString getMessage() const;

```

```
            level getLevel() const;
        };
    }
}

#endif // EVENTSERVERMESSAGE_H
```

## 8.121 C++/resources/events/eventuserchangenick.cpp File Reference

`#include "eventuserchangenick.h"`

Include dependency graph for eventuserchangenick.cpp:




### Namespaces

- namespace shared
- namespace shared::events

## 8.122 eventuserchangenick.cpp

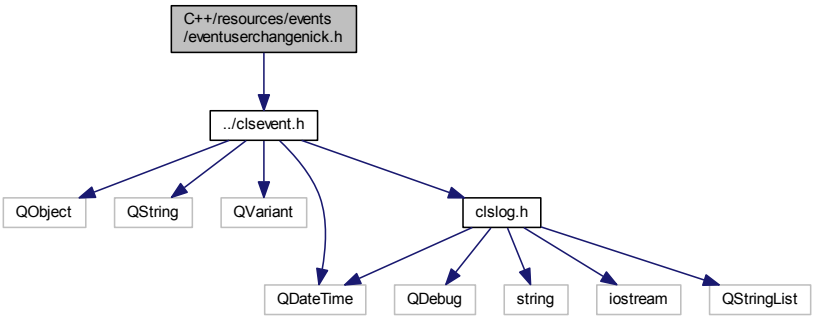
```
#include "eventuserchangenick.h"

namespace shared {
    namespace events {

        eventUserChangeNick::eventUserChangeNick(const int userId,
                                                 const QString newNick,
                                                 const QString oldNick) {
            ourType = EVENTTYPE_USER_NICK;
            this->addToData("userId",userId);
            this->addToData("newNick",newNick);
            this->addToData("oldNick",oldNick);
        }

        eventUserChangeNick::eventUserChangeNick(QVariantMap data)
            :clsEvent(data)
        { }

        int eventUserChangeNick::getUserId() const {
            return this->data["userId"].toInt();
        }

        QString eventUserChangeNick::getNewNick() const {
            return this->data["newNick"].toString();
        }

        QString eventUserChangeNick::getOldNick() const {
            return this->data["oldNick"].toString();
```

```
        }
    }
}
```

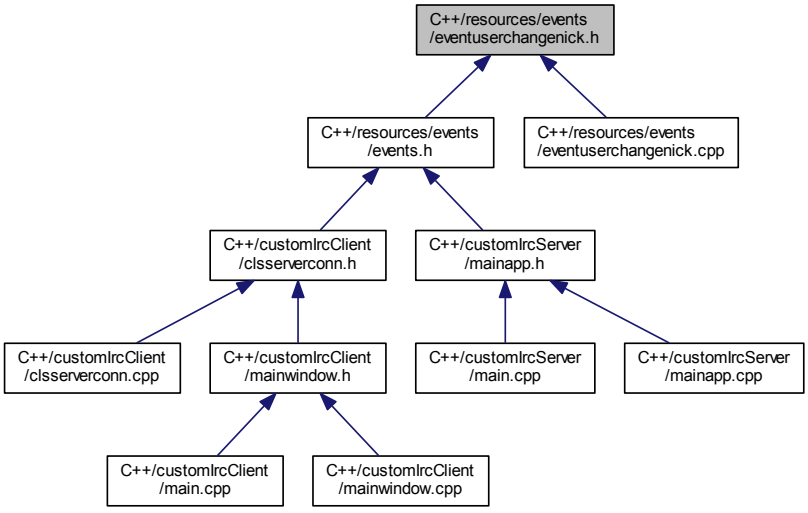
## 8.123 C++/resources/events/eventuserchangenick.h File Reference

`#include "../clsevent.h"`

Include dependency graph for eventuserchangenick.h:




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




### Classes

- class shared::events::eventUserChangeNick

  *Class containing a user who changes his nick-event (nick|status)*

### Namespaces

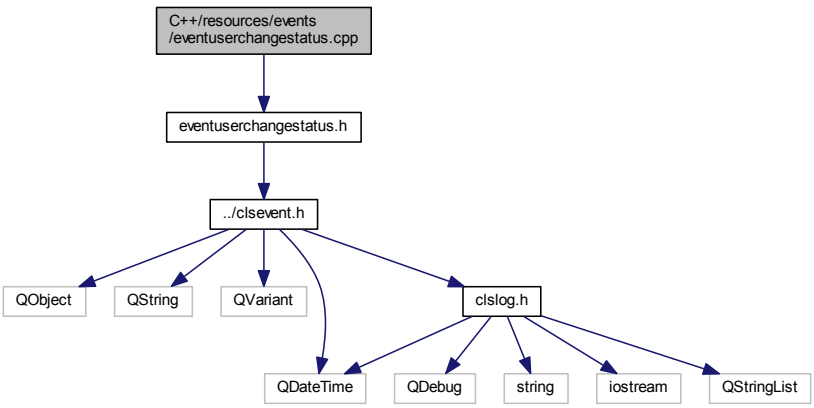
- namespace shared
- namespace shared::events

## 8.124 eventuserchangenick.h

```
00001 #ifndef EVENTUSERCHANGENICK_H
00002 #define EVENTUSERCHANGENICK_H
00003 #include "../clsevent.h"
00004
00005 namespace shared {
00006     namespace events {
00011         class eventUserChangeNick : public clsEvent
00012         {
00013         public:
00020             eventUserChangeNick(const int userId, const QString newNick,
00021                                 const QString oldNick);
00022
00028             eventUserChangeNick(QVariantMap data);
00029
00034             int getUserId() const;
00035
00040             QString getNewNick() const;
00041
00046             QString getOldNick() const;
00047         };
00048     }
00049 }
00050
00051 #endif // EVENTUSERCHANGENICK_H
```

## 8.125 C++/resources/events/eventuserchangestatus.cpp File Reference

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




### Namespaces

- namespace shared
- namespace shared::events

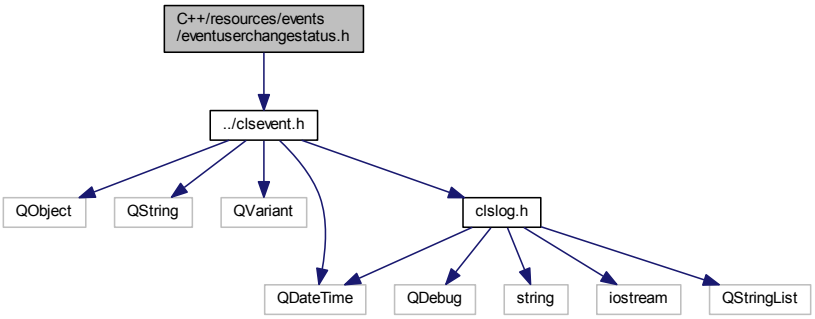
## 8.126 eventuserchangestatus.cpp

```
#include "eventuserchangestatus.h"

namespace shared {
    namespace events {

        eventUserChangeStatus::eventUserChangeStatus(const int userId,
                                                     const QString newStatus,
                                                     const QString oldStatus) {
            ourType = EVENTTYPE_USER_STATUS;
            this->addToData("userId",userId);
            this->addToData("newStatus",newStatus);
            this->addToData("oldStatus",oldStatus);
        }

        eventUserChangeStatus::eventUserChangeStatus(QVariantMap data)
            :clsEvent(data)
        { }

        int eventUserChangeStatus::getUserId() const {
            return this->data["userId"].toInt();
        }

        QString eventUserChangeStatus::getNewStatus() const {
            return this->data["newStatus"].toString();
        }

        QString eventUserChangeStatus::getOldStatus() const {
            return this->data["oldStatus"].toString();
        }
    }
}
```

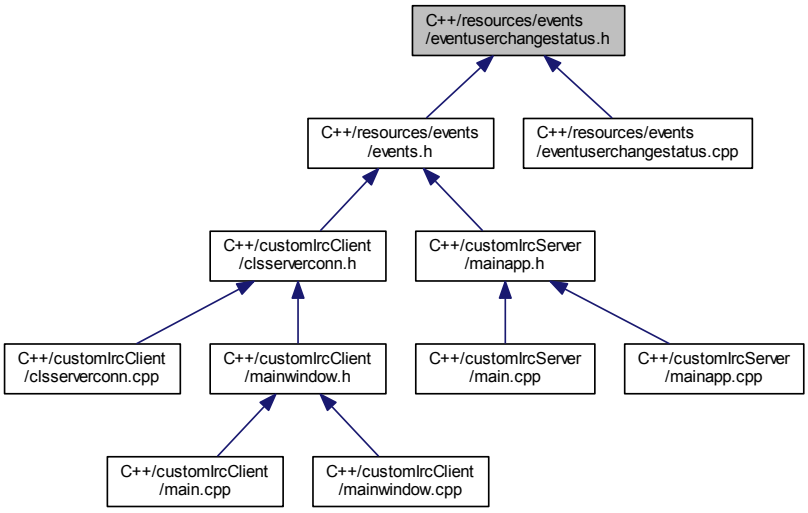
## 8.127 C++/resources/events/eventuserchangestatus.h File Reference

`#include "../clsevent.h"`

Include dependency graph for eventuserchangestatus.h:



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




## Classes

- class shared::events::eventUserChangeStatus

  *Class containing a user who changes his status-event (nick|status)*

## Namespaces

- namespace shared
- namespace shared::events

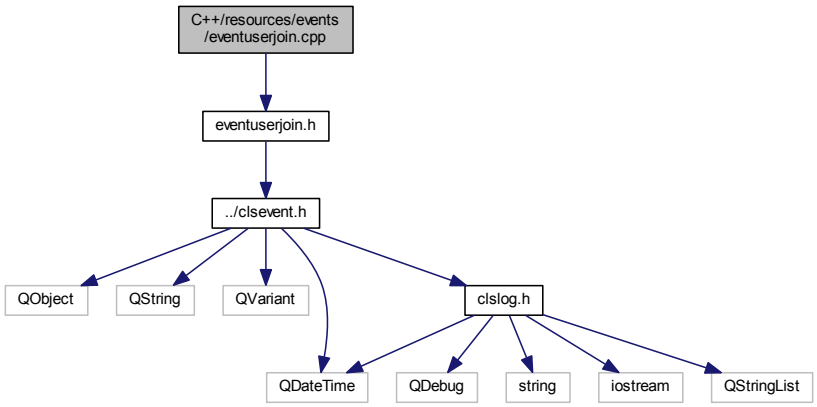
## 8.128 eventuserchangestatus.h

```
#ifndef EVENTUSERCHANGESTATUS_H
#define EVENTUSERCHANGESTATUS_H
#include "../clsevent.h"

namespace shared {
    namespace events {
        class eventUserChangeStatus : public clsEvent
        {
        public:
            eventUserChangeStatus(const int userId, const QString newStatus,
                                  const QString oldStatus);

            eventUserChangeStatus(QVariantMap data);

            int getUserId() const;

            QString getNewStatus() const;

            QString getOldStatus() const;
        };
    }
}


#endif // EVENTUSERCHANGESTATUS_H
```

## 8.129 C++/resources/events/eventuserjoin.cpp File Reference

`#include "eventuserjoin.h"`

Include dependency graph for eventuserjoin.cpp:




### Namespaces

- namespace shared
- namespace shared::events

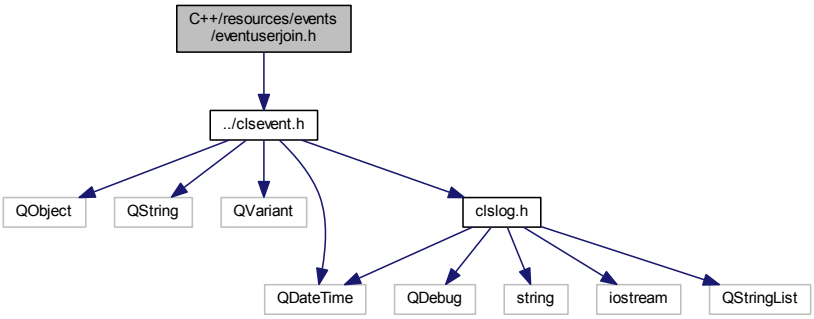
## 8.130 eventuserjoin.cpp

```
#include "eventuserjoin.h"

namespace shared{
    namespace events{
        eventUserJoin::eventUserJoin(const int userId, QVariantMap user){
            ourType = EVENTTYPE_USER_JOIN;
            this->addToData("userId", userId);
            this->addToData("user",QVariant(user));
        }

        eventUserJoin::eventUserJoin(QVariantMap data)
            :clsEvent(data)
        { }

        int eventUserJoin::getUserId() const{
            return this->getData("userId").toInt();
        }

        QVariantMap eventUserJoin::getUser() const{
            return this->getData("user").toMap();
        }
    }
}
```

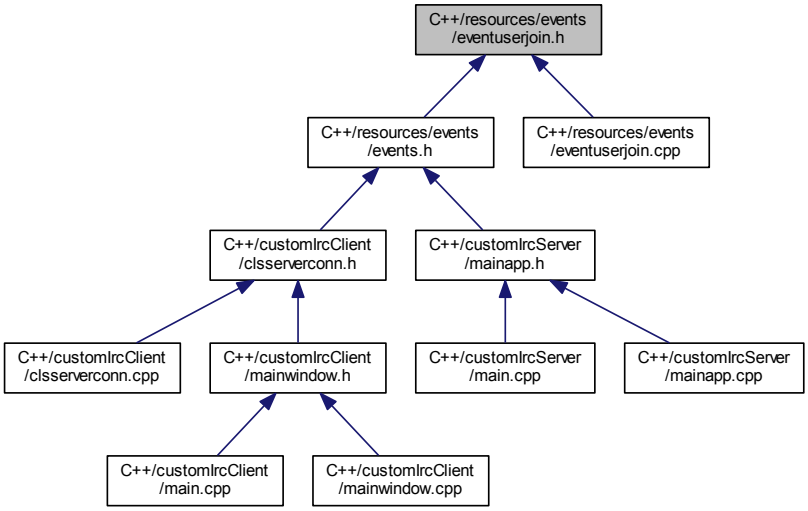
## 8.131 C++/resources/events/eventuserjoin.h File Reference

`#include "../clsevent.h"`

Include dependency graph for eventuserjoin.h:




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




## Classes

- class shared::events::eventUserJoin

  *Class containing a user JOIN event.*

## Namespaces

- namespace shared
- namespace shared::events

# 8.132 eventuserjoin.h

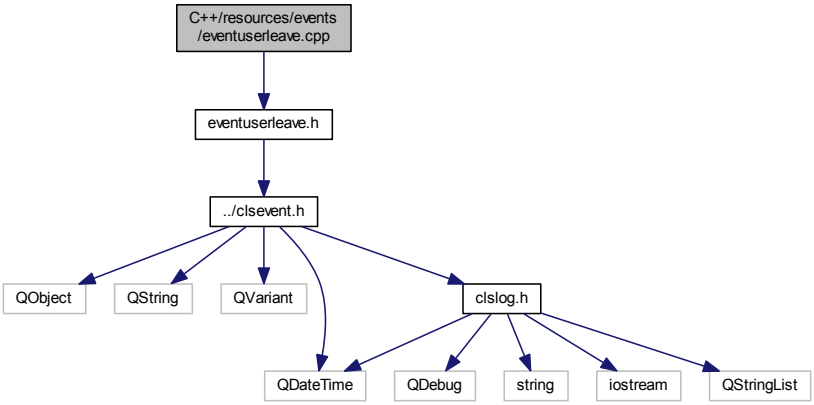
```
00001 #ifndef EVENTUSERJOIN_H
```

```
#define EVENTUSERJOIN_H
#include "../clsevent.h"

namespace shared{
    namespace events{
        class eventUserJoin : public clsEvent
        {
            public:
                eventUserJoin(const int userId, QVariantMap user);

                eventUserJoin(QVariantMap data);

                int getUserId() const;

                QVariantMap getUser() const;

        };
    }
}


#endif // EVENTUSERJOIN_H
```

## 8.133 C++/resources/events/eventuserleave.cpp File Reference

`#include "eventuserleave.h"`

Include dependency graph for eventuserleave.cpp:




### Namespaces

- namespace shared
- namespace shared::events

## 8.134 eventuserleave.cpp

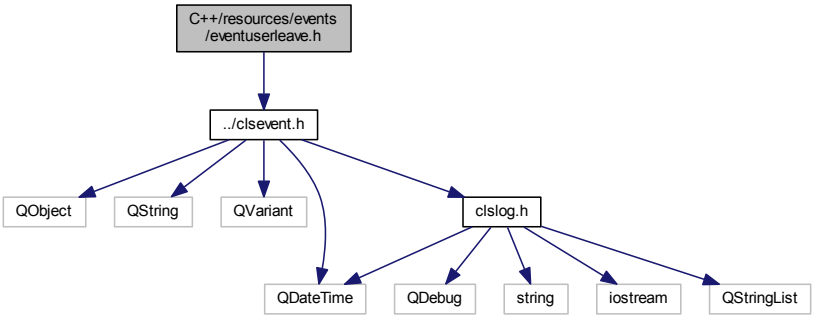
```
#include "eventuserleave.h"

namespace shared {
    namespace events {

        eventUserLeave::eventUserLeave(const int userId){
            ourType = EVENTTYPE_USER_LEFT;
            this->addToData("userId", userId);
        }

        eventUserLeave::eventUserLeave(QVariantMap data)
            :clsEvent(data)
```

```
        { }

        int eventUserLeave::getUserId() const{
            return this->getData("userId").toInt();
        }

    }
}
```

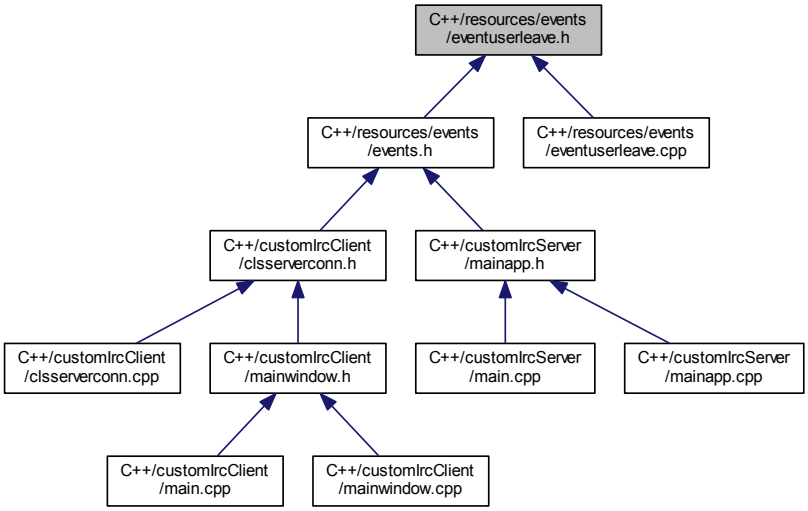
## 8.135 C++/resources/events/eventuserleave.h File Reference

`#include "../clsevent.h"`

Include dependency graph for eventuserleave.h:




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




### Classes

- class shared::events::eventUserLeave

  *Class containing a user PART event.*

### Namespaces

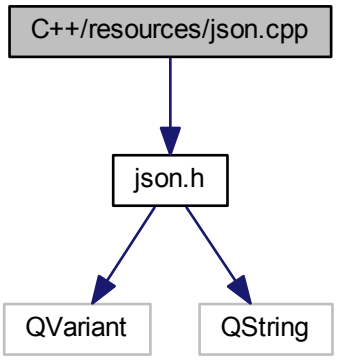
- namespace shared
- namespace shared::events

## 8.136 eventuserleave.h

```
#ifndef EVENTUSERLEAVE_H
#define EVENTUSERLEAVE_H
#include "../clsevent.h"

namespace shared {
    namespace events {
        class eventUserLeave : public clsEvent
        {
            public:
                eventUserLeave(const int userId);

                eventUserLeave(QVariantMap data);

                int getUserId() const;
        };

    }
}

#endif
```

## 8.137 C++/resources/json.cpp File Reference

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




### Namespaces

- namespace QtJson

  *A JSON data parser.*

### Enumerations

- enum QtJson::JsonToken {
  QtJson::JsonTokenNone = 0, QtJson::JsonTokenCurlyOpen = 1, QtJson::JsonTokenCurlyClose = 2, QtJson::JsonTokenSquaredOpen = 3,
  QtJson::JsonTokenSquaredClose = 4, QtJson::JsonTokenColon = 5, QtJson::JsonTokenComma = 6, QtJson::JsonTokenString = 7,
  QtJson::JsonTokenNumber = 8, QtJson::JsonTokenTrue = 9, QtJson::JsonTokenFalse = 10, QtJson::JsonTokenNull = 11 }

### Functions

- QVariant QtJson::parse (const QString &json)
- QVariant QtJson::parse (const QString &json, bool &success)
- QByteArray QtJson::serialize (const QVariant &data)
- QByteArray QtJson::serialize (const QVariant &data, bool &success)

## 8.138 json.cpp

```
00001 /* Copyright 2011 Eeli Reilin. All rights reserved.
00002  *
00003  * Redistribution and use in source and binary forms, with or without
00004  * modification, are permitted provided that the following conditions are met:
00005  *
00006  * 1. Redistributions of source code must retain the above copyright notice,
00007  *    this list of conditions and the following disclaimer.
00008  *
00009  * 2. Redistributions in binary form must reproduce the above copyright notice,
00010  *    this list of conditions and the following disclaimer in the documentation
00011  *    and/or other materials provided with the distribution.
00012  *
00013  * THIS SOFTWARE IS PROVIDED BY <COPYRIGHT HOLDER> ''AS IS'' AND ANY EXPRESS OR
00014  * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
00015  * MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
00016  * EVENT SHALL EELI REILIN OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
00017  * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
00018  * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA,
00019  * OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
00020  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
00021  * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
00022  * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00023  *
00024  * The views and conclusions contained in the software and documentation
00025  * are those of the authors and should not be interpreted as representing
00026  * official policies, either expressed or implied, of Eeli Reilin.
00027  */
00028
00033 #include "json.h"
00034
00035 namespace QtJson
00036 {
00037
00038
00039 static QString sanitizeString(QString str);
00040 static QByteArray join(const QList<QByteArray> &list, const QByteArray &sep);
00041 static QVariant parseValue(const QString &json, int &index, bool &success);
00042 static QVariant parseObject(const QString &json, int &index, bool &success);
00043 static QVariant parseArray(const QString &json, int &index, bool &success);
00044 static QVariant parseString(const QString &json, int &index, bool &success);
00045 static QVariant parseNumber(const QString &json, int &index);
00046 static int lastIndexOfNumber(const QString &json, int index);
00047 static void eatWhitespace(const QString &json, int &index);
00048 static int lookAhead(const QString &json, int index);
00049 static int nextToken(const QString &json, int &index);
00050
00051
00052
00053
00054 /***** public *****/
00055
00056
00060 QVariant parse(const QString &json)
00061 {
00062         bool success = true;
```

```
        return parse(json, success);
}

QVariant parse(const QString &json, bool &success)
{
        success = true;

        //Return an empty QVariant if the JSON data is either null or empty
        if(!json.isNull() || !json.isEmpty())
        {
                QString data = json;
                //We'll start from index 0
                int index = 0;

                //Parse the first value
                QVariant value = parseValue(data, index, success);

                //Return the parsed value
                return value;
        }
        else
        {
                //Return the empty QVariant
                return QVariant();
        }
}

QByteArray serialize(const QVariant &data)
{
        bool success = true;
        return serialize(data, success);
}

QByteArray serialize(const QVariant &data, bool &success)
{
        QByteArray str;
        success = true;

        if(!data.isValid()) // invalid or null?
        {
                str = "null";
        }
        else if((data.type() == QVariant::List) || (data.type() == QVariant::StringList)) // variant is a list?
        {
                QList<QByteArray> values;
                const QVariantList list = data.toList();
                Q_FOREACH(const QVariant& v, list)
                {
                        QByteArray serializedValue = serialize(v);
                        if(serializedValue.isNull())
                        {
                                success = false;
                                break;
                        }
                        values << serializedValue;
                }

                str = "[ " + join( values, ", " ) + " ]";
        }
        else if(data.type() == QVariant::Hash) // variant is a hash?
        {
            const QVariantHash vhash = data.toHash();
            QHashIterator<QString, QVariant> it( vhash );
            str = "{ ";
            QList<QByteArray> pairs;

            while(it.hasNext())
            {
                it.next();
                QByteArray serializedValue = serialize(it.value());

                if(serializedValue.isNull())
                {
                    success = false;
                    break;
                }

                pairs << sanitizeString(it.key()).toUtf8() + " : " + serializedValue;
            }

            str += join(pairs, ", ");
            str += " }";
        }
        else if(data.type() == QVariant::Map) // variant is a map?
        {
                const QVariantMap vmap = data.toMap();
```

```
                QMapIterator<QString, QVariant> it( vmap );
                str = "{ ";
                QList<QByteArray> pairs;
                while(it.hasNext())
                {
                        it.next();
                        QByteArray serializedValue = serialize(it.value());
                        if(serializedValue.isNull())
                        {
                                success = false;
                                break;
                        }
                        pairs << sanitizeString(it.key()).toUtf8() + " : " + serializedValue;
                }
                str += join(pairs, ", ");
                str += " }";
        }
        else if((data.type() == QVariant::String) || (data.type() == QVariant::ByteArray)) // a string or a byte array?
        {
                str = sanitizeString(data.toString()).toUtf8();
        }
        else if(data.type() == QVariant::Double) // double?
        {
                str = QByteArray::number(data.toDouble(), 'g', 20);
                if(!str.contains(".") && ! str.contains("e"))
                {
                        str += ".0";
                }
        }
        else if (data.type() == QVariant::Bool) // boolean value?
        {
                str = data.toBool() ? "true" : "false";
        }
        else if (data.type() == QVariant::ULongLong) // large unsigned number?
        {
                str = QByteArray::number(data.value<qulonglong>());
        }
        else if ( data.canConvert<qlonglong>() ) // any signed number?
        {
                str = QByteArray::number(data.value<qlonglong>());
        }
        else if (data.canConvert<long>())
        {
                str = QString::number(data.value<long>()).toUtf8();
        }
        else if (data.canConvert<QString>()) // can value be converted to string?
        {
                // this will catch QDate, QDateTime, QUrl, ...
                str = sanitizeString(data.toString()).toUtf8();
        }
        else
        {
                success = false;
        }
        if (success)
        {
                return str;
        }
        else
        {
                return QByteArray();
        }
}



/***** private *****/


enum JsonToken
{
        JsonTokenNone = 0,
        JsonTokenCurlyOpen = 1,
        JsonTokenCurlyClose = 2,
        JsonTokenSquaredOpen = 3,
        JsonTokenSquaredClose = 4,
        JsonTokenColon = 5,
        JsonTokenComma = 6,
        JsonTokenString = 7,
        JsonTokenNumber = 8,
        JsonTokenTrue = 9,
        JsonTokenFalse = 10,
        JsonTokenNull = 11
};

static QString sanitizeString(QString str)
```

```
{
        str.replace(QLatin1String("\\"), QLatin1String("\\\\"));
        str.replace(QLatin1String("\""), QLatin1String("\\\""));
        str.replace(QLatin1String("\b"), QLatin1String("\\b"));
        str.replace(QLatin1String("\f"), QLatin1String("\\f"));
        str.replace(QLatin1String("\n"), QLatin1String("\\n"));
        str.replace(QLatin1String("\r"), QLatin1String("\\r"));
        str.replace(QLatin1String("\t"), QLatin1String("\\t"));
        return QString(QLatin1String("\"%1\"")).arg(str);
}

static QByteArray join(const QList<QByteArray> &list, const QByteArray &sep)
{
        QByteArray res;
        Q_FOREACH(const QByteArray &i, list)
        {
                if(!res.isEmpty())
                {
                        res += sep;
                }
                res += i;
        }
        return res;
}

static QVariant parseValue(const QString &json, int &index, bool &success)
{
        //Determine what kind of data we should parse by
        //checking out the upcoming token
        switch(lookAhead(json, index))
        {
                case JsonTokenString:
                        return parseString(json, index, success);
                case JsonTokenNumber:
                        return parseNumber(json, index);
                case JsonTokenCurlyOpen:
                        return parseObject(json, index, success);
                case JsonTokenSquaredOpen:
                        return parseArray(json, index, success);
                case JsonTokenTrue:
                        nextToken(json, index);
                        return QVariant(true);
                case JsonTokenFalse:
                        nextToken(json, index);
                        return QVariant(false);
                case JsonTokenNull:
                        nextToken(json, index);
                        return QVariant();
                case JsonTokenNone:
                        break;
        }

        //If there were no tokens, flag the failure and return an empty QVariant
        success = false;
        return QVariant();
}

static QVariant parseObject(const QString &json, int &index, bool &success)
{
        QVariantMap map;
        int token;

        //Get rid of the whitespace and increment index
        nextToken(json, index);

        //Loop through all of the key/value pairs of the object
        bool done = false;
        while(!done)
        {
                //Get the upcoming token
                token = lookAhead(json, index);

                if(token == JsonTokenNone)
                {
                         success = false;
                         return QVariantMap();
                }
                else if(token == JsonTokenComma)
                {
                        nextToken(json, index);
                }
                else if(token == JsonTokenCurlyClose)
                {
                        nextToken(json, index);
                        return map;
                }
                else
```

```
00334                 {
00335                         //Parse the key/value pair's name
00336                         QString name = parseString(json, index, success).toString();
00337
00338                         if(!success)
00339                         {
00340                                 return QVariantMap();
00341                         }
00342
00343                         //Get the next token
00344                         token = nextToken(json, index);
00345
00346                         //If the next token is not a colon, flag the failure
00347                         //return an empty QVariant
00348                         if(token != JsonTokenColon)
00349                         {
00350                                 success = false;
00351                                 return QVariant(QVariantMap());
00352                         }
00353
00354                         //Parse the key/value pair's value
00355                         QVariant value = parseValue(json, index, success);
00356
00357                         if(!success)
00358                         {
00359                                 return QVariantMap();
00360                         }
00361
00362                         //Assign the value to the key in the map
00363                         map[name] = value;
00364                 }
00365         }
00366
00367         //Return the map successfully
00368         return QVariant(map);
00369 }
00370
00374 static QVariant parseArray(const QString &json, int &index, bool &success)
00375 {
00376         QVariantList list;
00377
00378         nextToken(json, index);
00379
00380         bool done = false;
00381         while(!done)
00382         {
00383                 int token = lookAhead(json, index);
00384
00385                 if(token == JsonTokenNone)
00386                 {
00387                         success = false;
00388                         return QVariantList();
00389                 }
00390                 else if(token == JsonTokenComma)
00391                 {
00392                         nextToken(json, index);
00393                 }
00394                 else if(token == JsonTokenSquaredClose)
00395                 {
00396                         nextToken(json, index);
00397                         break;
00398                 }
00399                 else
00400                 {
00401                         QVariant value = parseValue(json, index, success);
00402
00403                         if(!success)
00404                         {
00405                                 return QVariantList();
00406                         }
00407
00408                         list.push_back(value);
00409                 }
00410         }
00411
00412         return QVariant(list);
00413 }
00414
00418 static QVariant parseString(const QString &json, int &index, bool &success)
00419 {
00420         QString s;
00421         QChar c;
00422
00423         eatWhitespace(json, index);
00424
00425         c = json[index++];
00426
```

```
        bool complete = false;
        while(!complete)
        {
                if(index == json.size())
                {
                        break;
                }

                c = json[index++];

                if(c == '\"')
                {
                        complete = true;
                        break;
                }
                else if(c == '\\')
                {
                        if(index == json.size())
                        {
                                break;
                        }

                        c = json[index++];

                        if(c == '\"')
                        {
                                s.append('\"');
                        }
                        else if(c == '\\')
                        {
                                s.append('\\');
                        }
                        else if(c == '/')
                        {
                                s.append('/');
                        }
                        else if(c == 'b')
                        {
                                s.append('\b');
                        }
                        else if(c == 'f')
                        {
                                s.append('\f');
                        }
                        else if(c == 'n')
                        {
                                s.append('\n');
                        }
                        else if(c == 'r')
                        {
                                s.append('\r');
                        }
                        else if(c == 't')
                        {
                                s.append('\t');
                        }
                        else if(c == 'u')
                        {
                                int remainingLength = json.size() - index;

                                if(remainingLength >= 4)
                                {
                                        QString unicodeStr = json.mid(index, 4);

                                        int symbol = unicodeStr.toInt(0, 16);

                                        s.append(QChar(symbol));

                                        index += 4;
                                }
                                else
                                {
                                        break;
                                }
                        }
                }
                else
                {
                        s.append(c);
                }
        }

        if(!complete)
        {
                success = false;
                return QVariant();
        }
```

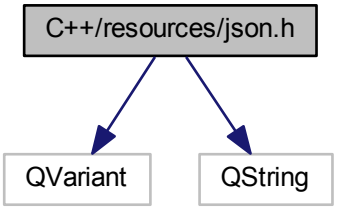
```
00514
00515         return QVariant(s);
00516 }
00517
00521 static QVariant parseNumber(const QString &json, int &index)
00522 {
00523         eatWhitespace(json, index);
00524
00525         int lastIndex = lastIndexOfNumber(json, index);
00526         int charLength = (lastIndex - index) + 1;
00527         QString numberStr;
00528
00529         numberStr = json.mid(index, charLength);
00530
00531         index = lastIndex + 1;
00532
00533         if (numberStr.contains('.')) {
00534                 return QVariant(numberStr.toDouble(NULL));
00535         } else if (numberStr.startsWith('-')) {
00536                 return QVariant(numberStr.toLongLong(NULL));
00537         } else {
00538                 return QVariant(numberStr.toULongLong(NULL));
00539         }
00540 }
00541
00545 static int lastIndexOfNumber(const QString &json, int index)
00546 {
00547         int lastIndex;
00548
00549         for(lastIndex = index; lastIndex < json.size(); lastIndex++)
00550         {
00551                 if(QString("0123456789+-.eE").indexOf(json[lastIndex]) == -1)
00552                 {
00553                         break;
00554                 }
00555         }
00556
00557         return lastIndex -1;
00558 }
00559
00563 static void eatWhitespace(const QString &json, int &index)
00564 {
00565         for(; index < json.size(); index++)
00566         {
00567                 if(QString(" \t\n\r").indexOf(json[index]) == -1)
00568                 {
00569                         break;
00570                 }
00571         }
00572 }
00573
00577 static int lookAhead(const QString &json, int index)
00578 {
00579         int saveIndex = index;
00580         return nextToken(json, saveIndex);
00581 }
00582
00586 static int nextToken(const QString &json, int &index)
00587 {
00588         eatWhitespace(json, index);
00589
00590         if(index == json.size())
00591         {
00592                 return JsonTokenNone;
00593         }
00594
00595         QChar c = json[index];
00596         index++;
00597         switch(c.toLatin1())
00598         {
00599                 case '{': return JsonTokenCurlyOpen;
00600                 case '}': return JsonTokenCurlyClose;
00601                 case '[': return JsonTokenSquaredOpen;
00602                 case ']': return JsonTokenSquaredClose;
00603                 case ',': return JsonTokenComma;
00604                 case '"': return JsonTokenString;
00605                 case '0': case '1': case '2': case '3': case '4':
00606                 case '5': case '6': case '7': case '8': case '9':
00607                 case '-': return JsonTokenNumber;
00608                 case ':': return JsonTokenColon;
00609         }
00610
00611         index--;
00612
00613         int remainingLength = json.size() - index;
00614
00615         //True
```

```
        if(remainingLength >= 4)
        {
                if (json[index] == 't' && json[index + 1] == 'r' &&
                        json[index + 2] == 'u' && json[index + 3] == 'e')
                {
                        index += 4;
                        return JsonTokenTrue;
                }
        }

        //False
        if (remainingLength >= 5)
        {
                if (json[index] == 'f' && json[index + 1] == 'a' &&
                        json[index + 2] == 'l' && json[index + 3] == 's' &&
                        json[index + 4] == 'e')
                {
                        index += 5;
                        return JsonTokenFalse;
                }
        }

        //Null
        if (remainingLength >= 4)
        {
                if (json[index] == 'n' && json[index + 1] == 'u' &&
                        json[index + 2] == 'l' && json[index + 3] == 'l')
                {
                        index += 4;
                        return JsonTokenNull;
                }
        }

        return JsonTokenNone;
}


} //end namespace
```

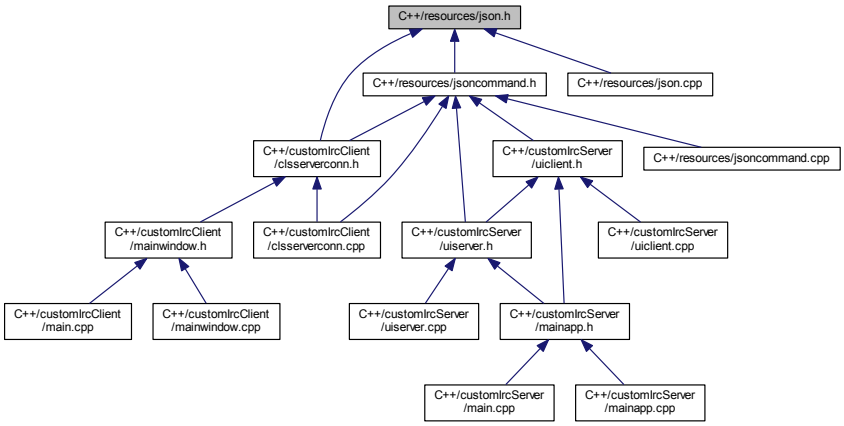
## 8.139 C++/resources/json.h File Reference

```
#include <QVariant>
#include <QString>
```

Include dependency graph for json.h:



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




## Namespaces

- namespace QtJson

  *A JSON data parser.*

## Functions

- QVariant QtJson::parse (const QString &json)
- QVariant QtJson::parse (const QString &json, bool &success)
- QByteArray QtJson::serialize (const QVariant &data)
- QByteArray QtJson::serialize (const QVariant &data, bool &success)

# 8.140 json.h

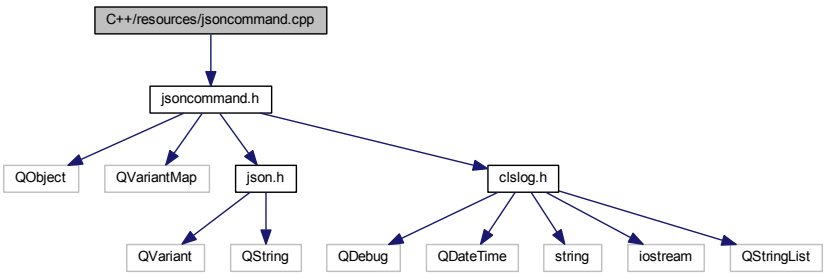
```
00001 /* Copyright 2011 Eeli Reilin. All rights reserved.
00002  *
00003  * Redistribution and use in source and binary forms, with or without
00004  * modification, are permitted provided that the following conditions are met:
00005  *
00006  * 1. Redistributions of source code must retain the above copyright notice,
00007  *    this list of conditions and the following disclaimer.
00008  *
00009  * 2. Redistributions in binary form must reproduce the above copyright notice,
00010  *    this list of conditions and the following disclaimer in the documentation
00011  *    and/or other materials provided with the distribution.
00012  *
00013  * THIS SOFTWARE IS PROVIDED BY <COPYRIGHT HOLDER> ''AS IS'' AND ANY EXPRESS OR
00014  * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
00015  * MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO
00016  * EVENT SHALL EELI REILIN OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
00017  * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
00018  * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA,
00019  * OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
00020  * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
00021  * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
00022  * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00023  *
00024  * The views and conclusions contained in the software and documentation
00025  * are those of the authors and should not be interpreted as representing
00026  * official policies, either expressed or implied, of Eeli Reilin.
00027  */
00028
00033 #ifndef JSON_H
00034 #define JSON_H
00035
```

```
#include <QVariant>
#include <QString>


namespace QtJson
{

    QVariant parse(const QString &json);

    QVariant parse(const QString &json, bool &success);

    QByteArray serialize(const QVariant &data);

    QByteArray serialize(const QVariant &data, bool &success);

} //end namespace

#endif //JSON_H
```

## 8.141 C++/resources/jsoncommand.cpp File Reference

`#include "jsoncommand.h"`

Include dependency graph for jsoncommand.cpp:




### Namespaces

- namespace K4U

  *Contains functions used for more projects.*

### Functions

- clsLog & K4U::operator<< (clsLog &log, jsonCommand &command)
- clsLog & K4U::operator<< (clsLog &log, const jsonCommand &command)

## 8.142 jsoncommand.cpp

```
#include "jsoncommand.h"

namespace K4U{

jsonCommand::jsonCommand(QString json){
    QVariant data = QtJson::parse(json);
    QVariantMap parent = qvariant_cast<QVariantMap>(data);
    this->setCommandInt(parent["command"].toString());
    this->data = parent["data"].toMap();
}


jsonCommand::jsonCommand(QVariantMap parent)
```

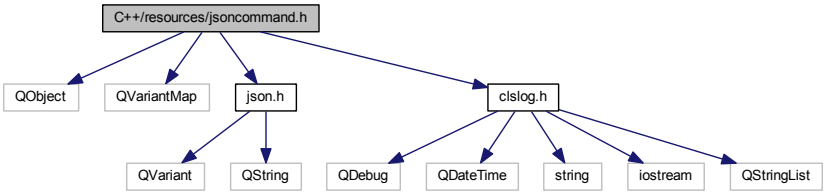
```
{
    this->command = (jsonCommands) parent["command"].toInt();
    this->data = parent["data"].toMap();
}

jsonCommand::jsonCommand(jsonCommands command)
    :command(command)
{ }


jsonCommands jsonCommand::getCommand() const{
    return this->command;
}

QString jsonCommand::getCommandString() const{
    QString ret = "";
    switch(this->command){
        case JSONCOMMAND_PING:
            ret = "PING";
            break;
        case JSONCOMMAND_EVENT:
            ret = "EVENT";
            break;
        case JSONCOMMAND_USERINFOREQ:
            ret = "USERINFOREQ";
            break;
        case JSONCOMMAND_USERQUERY:
            ret = "USERQUERY";
            break;
        case JSONCOMMAND_NONE:
            ret = "NONE";
            break;
        case JSONCOMMAND_OWNCHAT:
            ret = "OWNCHAT";
            break;
        case JSONCOMMAND_CHANGEOWNUSER:
            ret = "CHANGEOWNUSER";
            break;
        case JSONCOMMAND_GETEVENTS:
            ret = "GETEVENTS";
            break;
    }
    return ret;
}

void jsonCommand::setCommandInt(QString command){
    jsonCommands set = JSONCOMMAND_NONE;
    if(command == "PING"){
        set = JSONCOMMAND_PING;
    }else if(command == "EVENT"){
        set = JSONCOMMAND_EVENT;
    }else if(command == "USERINFOREQ"){
        set = JSONCOMMAND_USERINFOREQ;
    }else if(command == "USERQUERY"){
        set = JSONCOMMAND_USERQUERY;
    }else if(command == "OWNCHAT"){
        set = JSONCOMMAND_OWNCHAT;
    }else if(command == "CHANGEOWNUSER"){
        set = JSONCOMMAND_CHANGEOWNUSER;
    }else if(command == "GETEVENTS"){
        set = JSONCOMMAND_GETEVENTS;
    }
    this->command = set;
}


void jsonCommand::setCommand(const jsonCommands &newCommand){
    this->command = newCommand;
}


QVariantMap jsonCommand::getData() const{
    return this->data;
}

QVariant jsonCommand::getData(const QString key){
    return this->data[key];
}


void jsonCommand::setData(const QVariantMap &newData){
    this->data = newData;
}

void jsonCommand::addToData(const QString key, const QVariant &data){
    this->data.insert(key,data);
}
```

```
QString jsonCommand::toJsonString() const{
    QVariantMap vmRet;
    vmRet["command"] = this->getCommandString();
    vmRet["data"] = QVariant(this->getData());

    return QtJson::serialize(QVariant(vmRet));
}

clsLog& operator <<(clsLog &log, jsonCommand &command){
    log << "JSON command: " << command.getCommandString();
    log << " Data: " << endl << command.getData() << endl;
    return log;
}


clsLog& operator <<(clsLog &log, const jsonCommand &command){
    log << "JSON command: " << command.getCommandString();
    log << " Data: " << endl << command.getData() << endl;
    return log;
}

jsonCommand jsonCommand::operator =(
jsonCommand *other){
    this->command = other->command;
    this->data = other->data;
    return *this;
}

}
```

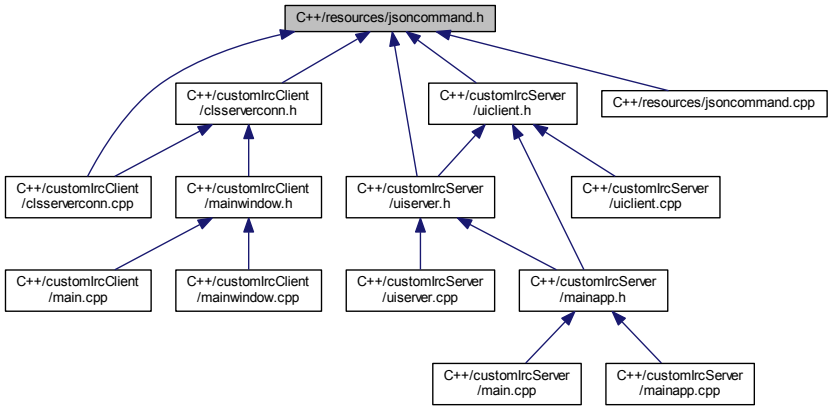
## 8.143 C++/resources/jsoncommand.h File Reference

```
#include <QObject>
#include <QVariantMap>
#include "json.h"
#include "clslog.h"
```

Include dependency graph for jsoncommand.h:



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




## Classes

- class K4U::jsonCommand

  *The jsonCommand class.*

## Namespaces

- namespace K4U

  *Contains functions used for more projects.*

## Enumerations

- enum K4U::jsonCommands { K4U::JSONCOMMAND_NONE = 0x00, K4U::JSONCOMMAND_PING = 0x01, K4U::JSONCOMMAND_EVENT, K4U::JSONCOMMAND_USERINFOREQ, K4U::JSONCOMMAND_USERQUERY, K4U::JSONCOMMAND_OWNCHAT, K4U::JSONCOMMAND_CHANGEOWNUSER, K4U::JSONCOMMAND_GETEVENTS }

# 8.144 jsoncommand.h

```
#ifndef JSONCOMMAND_H
#define JSONCOMMAND_H

#include <QObject>
#include <QVariantMap>

#include "json.h"
#include "clslog.h"

namespace K4U{
enum jsonCommands{
    JSONCOMMAND_NONE = 0x00,
    JSONCOMMAND_PING = 0x01,
    JSONCOMMAND_EVENT,
    JSONCOMMAND_USERINFOREQ,
    JSONCOMMAND_USERQUERY,
    JSONCOMMAND_OWNCHAT,
    JSONCOMMAND_CHANGEOWNUSER,
    JSONCOMMAND_GETEVENTS
};

```

```
00040 class jsonCommand{
00041 public:
00046     jsonCommand(QString json);
00052     jsonCommand(QVariantMap parent);
00057     jsonCommand(jsonCommands command);
00058
00063     jsonCommands getCommand() const;
00068     QString getCommandString() const;
00069
00070     void setCommand(const jsonCommands& newCommand);
00071     void setCommandInt(QString command);
00072
00078     QVariant getData(const QString key);
00079
00084     QVariantMap getData() const;
00085
00090     void setData(const QVariantMap& newData);
00091
00097     void addToData(const QString key, const QVariant &data);
00098
00099
00100     friend clsLog& operator<<(clsLog& log, jsonCommand &command);
00101     friend clsLog& operator<<(clsLog& log, const
      jsonCommand &command);
00102
00103     jsonCommand operator= (jsonCommand* other);
00104
00109     QString toJsonString() const;
00110 private:
00111     jsonCommands command;
00112     QVariantMap data;
00113 };
00114
00115 }
00116 #endif // JSONCOMMAND_H
```

# Index


~MainWindow
client::MainWindow, 219
MainWindow, 214
~chatBox
client::chatBox, 42
~chatEntry
client::chatEntry, 49
~chatNotification
client::chatNotification, 60
~clsUserEntry
client::clsUserEntry, 109
~clsUserList
client::clsUserList, 113
~ircClient
server::ircClient, 151
~ircUser
client::ircUser, 178
~uiServer
server::uiServer, 280

acceptConnection
server::uiServer, 280
add
server::ircUserList, 184
addChat
client::chatBox, 42, 43
addMessage
client::chatBox, 43, 44
addToData
K4U::jsonCommand, 191
shared::events::clsEvent, 65
addToEventHistory
server::mainApp, 200
addUser
client::clsUserList, 113

b
server::ircUser::userColor, 288
buffer
client::clsServerConn, 94
server::ircClient, 164
server::uiClient, 276

C++/customIrcClient/android/gen/R.java.d, 293, 294
C++/customIrcClient/android/gen/org/k4u/customIrc-Client/BuildConfig.java, 289
C++/customIrcClient/android/gen/org/k4u/customIrc-Client/R.java, 289, 290
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/-IMinistro.java, 290
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/-IMinistro.java.d, 292
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/-IMinistroCallback.java, 292
C++/customIrcClient/android/gen/org/kde/necessitas/ministro/-IMinistroCallback.java.d, 293
C++/customIrcClient/android/main.cpp, 294, 295
C++/customIrcClient/android/mainwindow.cpp, 297, 298
C++/customIrcClient/android/mainwindow.h, 303
C++/customIrcClient/android/src/org/kde/necessitas/origo/-QtActivity.java, 306, 307
C++/customIrcClient/android/src/org/kde/necessitas/origo/-QtApplication.java, 323
C++/customIrcClient/chatbox.cpp, 325, 326
C++/customIrcClient/chatbox.h, 327, 328
C++/customIrcClient/chatentry.cpp, 329
C++/customIrcClient/chatentry.h, 331, 332
C++/customIrcClient/chatinput.cpp, 332, 333
C++/customIrcClient/chatinput.h, 333, 334
C++/customIrcClient/chatnotification.cpp, 335
C++/customIrcClient/chatnotification.h, 336, 337
C++/customIrcClient/clsserverconn.cpp, 338
C++/customIrcClient/clsserverconn.h, 340, 342
C++/customIrcClient/clssettings.cpp, 343
C++/customIrcClient/clssettings.h, 346, 347
C++/customIrcClient/desktop/clsnotify.cpp, 350
C++/customIrcClient/desktop/clsnotify.h, 351
C++/customIrcClient/desktop/clsuserentry.cpp, 352
C++/customIrcClient/desktop/clsuserentry.h, 353, 354
C++/customIrcClient/desktop/clsuserlist.cpp, 355
C++/customIrcClient/desktop/clsuserlist.h, 356, 357
C++/customIrcClient/ircuser.cpp, 357, 358
C++/customIrcClient/ircuser.h, 364
C++/customIrcClient/main.cpp, 295, 296
C++/customIrcClient/mainwindow.cpp, 299
C++/customIrcClient/mainwindow.h, 304, 305
C++/customIrcServer/clssettings.cpp, 344
C++/customIrcServer/clssettings.h, 348, 349
C++/customIrcServer/functions.cpp, 368
C++/customIrcServer/functions.h, 369, 370
C++/customIrcServer/ircCodes.h, 379, 383
C++/customIrcServer/ircclient.cpp, 371
C++/customIrcServer/ircclient.h, 375, 377
C++/customIrcServer/ircuser.cpp, 359, 360
C++/customIrcServer/ircuser.h, 365, 366
C++/customIrcServer/main.cpp, 296, 297
C++/customIrcServer/mainapp.cpp, 386
C++/customIrcServer/mainapp.h, 390, 391
C++/customIrcServer/uiclient.cpp, 392

C++/customIrcServer/uiclient.h, 393, 394
C++/customIrcServer/uiserver.cpp, 395, 396
C++/customIrcServer/uiserver.h, 397, 398
C++/resources/clsevent.cpp, 399
C++/resources/clsevent.h, 400, 401
C++/resources/clslog.cpp, 402, 403
C++/resources/clslog.h, 405, 406
C++/resources/events/eventchat.cpp, 406, 407
C++/resources/events/eventchat.h, 407, 408
C++/resources/events/events.h, 409, 410
C++/resources/events/eventservermessage.cpp, 410, 411
C++/resources/events/eventservermessage.h, 411, 412
C++/resources/events/eventuserchangenick.cpp, 413
C++/resources/events/eventuserchangenick.h, 414, 415
C++/resources/events/eventuserchangestatus.cpp, 415, 416
C++/resources/events/eventuserchangestatus.h, 416, 417
C++/resources/events/eventuserjoin.cpp, 418
C++/resources/events/eventuserjoin.h, 418, 419
C++/resources/events/eventuserleave.cpp, 420
C++/resources/events/eventuserleave.h, 421, 422
C++/resources/json.cpp, 422, 423
C++/resources/json.h, 430, 431
C++/resources/jsoncommand.cpp, 432
C++/resources/jsoncommand.h, 434, 435
calculateMessageWidth
    client::chatEntry, 49
    client::chatNotification, 60
cb
    client::MainWindow, 229
changeOwnUser
    server::uiClient, 272
    server::uiServer, 280
channel
    server::clsSettings, 106
    server::ircClient, 164
chatBox
    client::chatBox, 42
chatEntry
    client::chatEntry, 48, 49
chatInput
    client::chatInput, 54
chatNotification
    client::chatNotification, 59
chatReceived
    client::clsServerConn, 83
    client::MainWindow, 219
    server::ircClient, 151
    server::mainApp, 200
    server::uiClient, 272
    server::uiServer, 281
checkForCommand
    client::chatInput, 54
checkForJsonCommand
    client::clsServerConn, 83
    server::uiClient, 273
checkForPing
    server::ircClient, 152
clear
    client::clsUserList, 113
client, 11
client::chatBox
    WIDGETTYPES_CHATENTRY, 42
    WIDGETTYPES_NOTIFICATION, 42
client::MainWindow, 215
    ~MainWindow, 219
    cb, 229
    chatReceived, 219
    commandGiven, 220
    connectedTimeout, 221
    doReconnect, 221
    log, 229
    MainWindow, 218
    moveScrollBarToBottom, 222
    reconnectTimer, 229
    sendChat, 222
    server, 229
    serverConnected, 223
    serverDisconnected, 224
    serverMessageReceived, 224
    settings, 229
    showMessage, 225
    ui, 230
    userChangeNick, 225
    userEnter, 226
    userLeave, 227
    userQueryCompleted, 228
    userStatusChange, 228
    users, 230
client::chatBox, 40
    ~chatBox, 42
    addChat, 42, 43
    addMessage, 43, 44
    chatBox, 42
    entries, 45
    setAllUserNameSize, 45
    ui, 45
    widgetTypes, 42
    widthOfNick, 45
client::chatBox::content, 115
    type, 115
    widget, 115
client::chatEntry, 46
    ~chatEntry, 49
    calculateMessageWidth, 49
    chatEntry, 48, 49
    getHeight, 50
    getWidth, 50
    init, 50
    message, 52
    name, 52
    resizeEvent, 51
    setUserWidth, 51
    timeOfMessage, 52

ui, 52
client::chatInput, 53
chatInput, 54
checkForCommand, 54
commandGiven, 55
keyPressEvent, 55
sendChat, 55
client::chatNotification, 56
~chatNotification, 60
calculateMessageWidth, 60
chatNotification, 59
getHeight, 61
getWidth, 61
message, 62
resizeEvent, 61
timeOfMessage, 62
ui, 62
client::clsNotify, 76
clsNotify, 78
show, 78
client::clsServerConn, 78
buffer, 94
chatReceived, 83
checkForJsonCommand, 83
clsServerConn, 82
connectTimeout, 84
connectTimer, 94
connected, 84
depth, 94
disconnected, 85
doConnect, 85
handleCompletedQuery, 85
handleEventReceived, 86
handleEventRequest, 87
handleOwnUserChange, 88
hostName, 94
isConnected, 94
isConnecting, 94
log, 95
port, 95
readData, 89
sendCommand, 90
serverMessageReceived, 91
sgnConnectTimeout, 91
sock, 95
srvConnected, 91
srvDisconnected, 92
userChangeNick, 92
userEnter, 93
userLeave, 93
userQueryCompleted, 93
userStatusChange, 94
client::clsSettings, 95
clsSettings, 96
getPort, 96
getServer, 97
port, 97
readServer, 97
server, 98
client::clsUserEntry, 107
~clsUserEntry, 109
clsUserEntry, 109
iconChanged, 109
refresh, 110
ui, 110
user, 111
client::clsUserList, 111
~clsUserList, 113
addUser, 113
clear, 113
clsUserList, 113
refreshUser, 114
removeUser, 114
setupUi, 114
users, 115
verticalLayout, 115
verticalSpacer, 115
client::ircUser, 175
~ircUser, 178
copyFrom, 178
email, 182
getColor, 178
getEmail, 179
getIcon, 179
getId, 179
getNick, 179
getOnline, 180
getStandard, 180
getStatus, 180
icon, 182
iconChanged, 181
id, 182
ircUser, 177, 178
log, 182
nick, 182
online, 182
setColor, 181
setEmail, 181
setNick, 181
setOnline, 181
setStatus, 181
slot_netwManagerFinished, 181
standard, 182
status, 183
uColor, 183
clsEvent
shared::events::clsEvent, 64
clsLog
K4U::clsLog, 70
clsNotify
client::clsNotify, 78
clsServerConn
client::clsServerConn, 82
clsSettings
client::clsSettings, 96
server::clsSettings, 100

clsUserEntry
client::clsUserEntry, 109
clsUserList
client::clsUserList, 113
command
K4U::jsonCommand, 196
commandGiven
client::chatInput, 55
client::MainWindow, 220
connectTimeout
client::clsServerConn, 84
server::ircClient, 152
connectTimer
client::clsServerConn, 94
server::ircClient, 164
connected
client::clsServerConn, 84
server::ircClient, 164
connectedTimeout
client::MainWindow, 221
connecting
server::ircClient, 164
connections
server::uiServer, 286
copyFrom
client::ircUser, 178
customIrcClient/android/main.cpp
main, 295
customIrcClient/main.cpp
main, 296
customIrcServer/main.cpp
main, 296

DEBUG
org::k4u::customIrcClient::BuildConfig, 39
DEX_PATH_KEY
org::kde::necessitas::origo::QtActivity, 258
data
K4U::jsonCommand, 196
shared::events::clsEvent, 68
del
server::ircUserList, 185
depth
client::clsServerConn, 94
server::uiClient, 276
disconnected
client::clsServerConn, 85
server::ircClient, 153
server::uiClient, 273
dispatchGenericMotionEvent
org::kde::necessitas::origo::QtApplication, 266
dispatchKeyEvent
org::kde::necessitas::origo::QtActivity, 236
org::kde::necessitas::origo::QtApplication, 266
dispatchKeyShortcutEvent
org::kde::necessitas::origo::QtApplication, 266
dispatchPopulateAccessibilityEvent
org::kde::necessitas::origo::QtActivity, 236
org::kde::necessitas::origo::QtApplication, 266
dispatchTouchEvent
org::kde::necessitas::origo::QtActivity, 237
org::kde::necessitas::origo::QtApplication, 266
dispatchTrackballEvent
org::kde::necessitas::origo::QtActivity, 237
org::kde::necessitas::origo::QtApplication, 266
doConnect
client::clsServerConn, 85
server::ircClient, 153
doEndl
K4U::clsLog, 71
doIrcReconnect
server::mainApp, 201
doReconnect
client::MainWindow, 221
doSendEvents
server::mainApp, 202
server::uiServer, 282
doUserQuery
server::mainApp, 202
server::uiServer, 282
downloadUpgradeMinistro
org::kde::necessitas::origo::QtActivity, 237

ERR_ACCEPTEXIST
ircCodes, 21
ERR_ACCEPTFULL
ircCodes, 21
ERR_ACCEPTNOT
ircCodes, 21
ERR_ACCOUNTONLY
ircCodes, 22
ERR_ADMONLY
ircCodes, 23
ERR_ALREADYREGISTERED
ircCodes, 21
ERR_ATTACKDENY
ircCodes, 22
ERR_BADCHANMASK
ircCodes, 21
ERR_BADCHANNAME
ircCodes, 21
ERR_BADCHANNELKEY
ircCodes, 21
ERR_BADCHARENCODING
ircCodes, 24
ERR_BADEXPIRE
ircCodes, 22
ERR_BADFEATURE
ircCodes, 22
ERR_BADHOSTMASK
ircCodes, 23
ERR_BADLOGSYS
ircCodes, 22
ERR_BADLOGTYPE
ircCodes, 22
ERR_BADLOGVALUE
ircCodes, 22
ERR_BADMASK

ircCodes, 20
ERR_BADPING
ircCodes, 22
ERR_BANLISTFULL
ircCodes, 21
ERR_BANNEDFROMCHAN
ircCodes, 21
ERR_BANNICKCHANGE
ircCodes, 21
ERR_BANONCHAN
ircCodes, 20
ERR_CANNOTCHANGECHANMODE
ircCodes, 24
ERR_CANNOTCHANGESERVERMODE
ircCodes, 24
ERR_CANNOTCHANGEUMODE
ircCodes, 24
ERR_CANNOTDOCOMMAND
ircCodes, 24
ERR_CANNOTKNOCK
ircCodes, 22
ERR_CANNOTSENDTOCHAN
ircCodes, 20
ERR_CANNOTSENDTONICK
ircCodes, 24
ERR_CANTKICKADMIN
ircCodes, 22
ERR_CANTKILLSERVER
ircCodes, 22
ERR_CHANNELISFULL
ircCodes, 21
ERR_CHANOPEN
ircCodes, 24
ERR_CHANOPRIVSNEEDED
ircCodes, 22
ERR_CHANOWNPRIVNEEDED
ircCodes, 22
ERR_CHANTOORECENT
ircCodes, 22
ERR_DEAD
ircCodes, 21
ERR_DESYNC
ircCodes, 22
ERR_DISABLED
ircCodes, 22
ERR_DONTCHEAT
ircCodes, 22
ERR_ERRONEUSNICKNAME
ircCodes, 20
ERR_EVENTNICKCHANGE
ircCodes, 20
ERR_FILEERROR
ircCodes, 20
ERR_GHOSTEDCLIENT
ircCodes, 22
ERR_HOSTILENAME
ircCodes, 21
ERR_HOSTUNAVAIL
ircCodes, 23
ERR_HTMDISABLED
ircCodes, 22
ERR_IDCOLLISION
ircCodes, 21
ERR_INVALID_ERROR
ircCodes, 22
ERR_INVALIDUSERNAME
ircCodes, 21
ERR_INVITEONLYCHAN
ircCodes, 21
ERR_ISCHANSERVICE
ircCodes, 22
ERR_ISOPERLCHAN
ircCodes, 22
ERR_ISREALSERVICE
ircCodes, 22
ERR_KEYSET
ircCodes, 21
ERR_KICKEDFROMCHAN
ircCodes, 21
ERR_KILLDENY
ircCodes, 22
ERR_KNOCKDISABLED
ircCodes, 24
ERR_KNOCKONCHAN
ircCodes, 24
ERR_LENGTHTRUNCATED
ircCodes, 20
ERR_LINKCHANNEL
ircCodes, 21
ERR_LINKFAIL
ircCodes, 21
ERR_LINKSET
ircCodes, 21
ERR_LISTSYNTAX
ircCodes, 23
ERR_LONGMASK
ircCodes, 22
ERR_MASKTOOWIDE
ircCodes, 23
ERR_MSGSERVICES
ircCodes, 22
ERR_NEEDMOREPARAMS
ircCodes, 21
ERR_NEEDREGGEDNICK
ircCodes, 21
ERR_NICKCOLLISION
ircCodes, 20
ERR_NICKLOST
ircCodes, 21
ERR_NICKNAMEINUSE
ircCodes, 20
ERR_NICKTOOFAST
ircCodes, 21
ERR_NOADMININFO
ircCodes, 20
ERR_NOCHANMODES

ircCodes, 21
ERR_NOCOLORSONCHAN
ircCodes, 20
ERR_NOFEATURE
ircCodes, 22
ERR_NOHIDING
ircCodes, 21
ERR_NOINVITE
ircCodes, 22
ERR_NOLANGUAGE
ircCodes, 24
ERR_NOLOGIN
ircCodes, 21
ERR_NOMOTD
ircCodes, 20
ERR_NONICKCHANGE
ircCodes, 21
ERR_NONICKNAMEGIVEN
ircCodes, 20
ERR_NONONREG
ircCodes, 22
ERR_NOOPERHOST
ircCodes, 22
ERR_NOOPERMOTD
ircCodes, 20
ERR_NOORIGIN
ircCodes, 20
ERR_NOPERMFORHOST
ircCodes, 21
ERR_NOPRIVILEGES
ircCodes, 22
ERR_NOPRIVS
ircCodes, 24
ERR_NORECIPIENT
ircCodes, 20
ERR_NORULES
ircCodes, 20
ERR_NOSERVICEHOST
ircCodes, 22
ERR_NOSUCHCHANNEL
ircCodes, 20
ERR_NOSUCHNICK
ircCodes, 20
ERR_NOSUCHSERVER
ircCodes, 20
ERR_NOSUCHSERVICE
ircCodes, 20
ERR_NOTEXTTOSEND
ircCodes, 20
ERR_NOTFORHALFOPS
ircCodes, 21
ERR_NOTIMPLEMENTED
ircCodes, 21
ERR_NOTONCHANNEL
ircCodes, 21
ERR_NOTOPLEVEL
ircCodes, 20
ERR_NOTREGISTERED
ircCodes, 21
ERR_NOULINE
ircCodes, 22
ERR_NUMERIC_ERR
ircCodes, 24
ERR_ONLYSERVERSCANCHANGE
ircCodes, 21
ERR_OPERONLY
ircCodes, 23
ERR_OPERSPVERIFY
ircCodes, 23
ERR_PASSWDMISMATCH
ircCodes, 21
ERR_PFXUNROUTABLE
ircCodes, 23
ERR_QUARANTINED
ircCodes, 23
ERR_QUERYTOOLONG
ircCodes, 20
ERR_REMOTEPFX
ircCodes, 23
ERR_RESTRICTED
ircCodes, 22
ERR_SECUREONLYCHAN
ircCodes, 22
ERR_SERVERMODELOCK
ircCodes, 24
ERR_SERVICECONFUSED
ircCodes, 20
ERR_SERVICENAMEINUSE
ircCodes, 20
ERR_SERVICESDOWN
ircCodes, 21
ERR_SILELISTFULL
ircCodes, 22
ERR_STATSSLINE
ircCodes, 23
ERR_SUMMONDISABLED
ircCodes, 21
ERR_TARGETTOOFAST
ircCodes, 21
ERR_TEXTTOOSHORT
ircCodes, 24
ERR_TOOMANYAWAY
ircCodes, 20
ERR_TOOMANYCHANNELS
ircCodes, 20
ERR_TOOMANYDCC
ircCodes, 22
ERR_TOOMANYKNOCK
ircCodes, 24
ERR_TOOMANYLANGUAGES
ircCodes, 24
ERR_TOOMANYMATCHES
ircCodes, 20
ERR_TOOMANYTARGETS
ircCodes, 20
ERR_TOOMANYUSERS

ircCodes, 23
ERR_TOOMANYWATCH
ircCodes, 22
ERR_TSLESSCHAN
ircCodes, 22
ERR_UMODEUNKNOWNFLAG
ircCodes, 22
ERR_UNAVAILRESOURCE
ircCodes, 20
ERR_UNIQOPRIVSNEEDED
ircCodes, 22
ERR_UNKNOWNCOMMAND
ircCodes, 20
ERR_UNKNOWNERROR
ircCodes, 20
ERR_UNKNOWNMODE
ircCodes, 21
ERR_UNKNOWNSERVERMODE
ircCodes, 24
ERR_USERNOTINCHANNEL
ircCodes, 21
ERR_USERNOTONSERV
ircCodes, 22
ERR_USERONCHANNEL
ircCodes, 21
ERR_USERSDISABLED
ircCodes, 21
ERR_USERSDONTMATCH
ircCodes, 22
ERR_USINGSLINE
ircCodes, 23
ERR_VOICENEEDED
ircCodes, 22
ERR_VWORLDWARN
ircCodes, 22
ERR_WASNOSUCHNICK
ircCodes, 20
ERR_WHOLIMEXCEED
ircCodes, 23
ERR_WHOSYNTAX
ircCodes, 23
ERR_WHOTRUNC
ircCodes, 23
ERR_WILDTOPLEVEL
ircCodes, 20
ERR_YOUREBANNEDCREEP
ircCodes, 21
ERR_YOUWILLBEBANNED
ircCodes, 21
EVENTTYPE_CHAT
shared::events, 37
EVENTTYPE_NONE
shared::events, 37
EVENTTYPE_SERVERMSG
shared::events, 37
EVENTTYPE_USER_JOIN
shared::events, 37
EVENTTYPE_USER_LEFT
shared::events, 37
EVENTTYPE_USER_NICK
shared::events, 37
EVENTTYPE_USER_STATUS
shared::events, 37
ERROR_CODE_KEY
org::kde::necessitas::origo::QtActivity, 258
email
client::ircUser, 182
server::ircUser, 174
endl
K4U, 26
entries
client::chatBox, 45
eventChat
shared::events::eventChat, 118
eventServerMessage
shared::events::eventServerMessage, 122
eventTypes
shared::events, 37
eventUserChangeNick
shared::events::eventUserChangeNick, 127
eventUserChangeStatus
shared::events::eventUserChangeStatus, 132
eventUserJoin
shared::events::eventUserJoin, 136
eventUserLeave
shared::events::eventUserLeave, 140
events
server::mainApp, 211

findUser
server::ircUserList, 186

g
server::ircUser::userColor, 288
getAll
server::ircUserList, 186
getChannel
server::clsSettings, 101
getColor
client::ircUser, 178
server, 33
getCommand
K4U::jsonCommand, 192
getCommandString
K4U::jsonCommand, 192
getData
K4U::jsonCommand, 193
shared::events::clsEvent, 65
getEmail
client::ircUser, 179
getEventType
shared::events::clsEvent, 66
getEvents
server::uiClient, 274
getHeight
client::chatEntry, 50
client::chatNotification, 61

getIcon
client::ircUser, 179
getId
client::ircUser, 179
server, 33
server::ircUser, 169
getIrcId
server::ircUser, 170
getLevel
shared::events::eventServerMessage, 122
getListeningPort
server::clsSettings, 101
getMessage
shared::events::eventChat, 118
shared::events::eventServerMessage, 123
getName
server::ircUser, 170
getNewNick
shared::events::eventUserChangeNick, 128
getNewStatus
shared::events::eventUserChangeStatus, 133
getNick
client::ircUser, 179
getNickAndStatus
server, 33
getNickAndStatusFromId
server, 34
getOldNick
shared::events::eventUserChangeNick, 128
getOldStatus
shared::events::eventUserChangeStatus, 133
getOnline
client::ircUser, 180
server::ircUser, 170
getOwnNick
server::ircClient, 154
getOwnUser
server::clsSettings, 102
getPort
client::clsSettings, 96
getServer
client::clsSettings, 97
server::clsSettings, 102
getStandard
client::ircUser, 180
getStatus
client::ircUser, 180
server::ircUser, 171
getTag
K4U::clsLog, 72
getTime
shared::events::clsEvent, 67
getType
shared::events::clsEvent, 67
getUser
server::ircUserList, 186
shared::events::eventChat, 119
shared::events::eventUserJoin, 136
getUserById
server::ircUserList, 187
getUserByIrcId
server::ircUserList, 187
getUserByNick
server::ircUserList, 188
getUserId
shared::events::eventUserChangeNick, 128
shared::events::eventUserChangeStatus, 133
shared::events::eventUserJoin, 137
shared::events::eventUserLeave, 140
getUserName
server::clsSettings, 103
getUsers
server::clsSettings, 103
getWidth
client::chatEntry, 50
client::chatNotification, 61

handleCommand
server::ircClient, 154
handleCompletedQuery
client::clsServerConn, 85
handleEventReceived
client::clsServerConn, 86
handleEventRequest
client::clsServerConn, 87
handleOwnUserChange
client::clsServerConn, 88
hostName
client::clsServerConn, 94
hostname
server::ircClient, 165

IRCConnectTimeOut
server::ircClient, 157
IRCConnected
server::ircClient, 156
IRCDisconnected
server::ircClient, 157
icon
client::ircUser, 182
iconChanged
client::clsUserEntry, 109
client::ircUser, 181
id
client::ircUser, 182
server::ircUser, 174
inCommand
server, 34
init
client::chatEntry, 50
server::ircClient, 155
invokeDelegate
org::kde::necessitas::origo::QtApplication, 263
invokeDelegateMethod
org::kde::necessitas::origo::QtApplication, 264
irc
server::mainApp, 211

ircCodes
ERR_ACCEPTEXIST, 21
ERR_ACCEPTFULL, 21
ERR_ACCEPTNOT, 21
ERR_ACCOUNTONLY, 22
ERR_ADMONLY, 23
ERR_ALREADYREGISTERED, 21
ERR_ATTACKDENY, 22
ERR_BADCHANMASK, 21
ERR_BADCHANNAME, 21
ERR_BADCHANNELKEY, 21
ERR_BADCHARENCODING, 24
ERR_BADEXPIRE, 22
ERR_BADFEATURE, 22
ERR_BADHOSTMASK, 23
ERR_BADLOGSYS, 22
ERR_BADLOGTYPE, 22
ERR_BADLOGVALUE, 22
ERR_BADMASK, 20
ERR_BADPING, 22
ERR_BANLISTFULL, 21
ERR_BANNEDFROMCHAN, 21
ERR_BANNICKCHANGE, 21
ERR_BANONCHAN, 20
ERR_CANNOTCHANGECHANMODE, 24
ERR_CANNOTCHANGESERVERMODE, 24
ERR_CANNOTCHANGEUMODE, 24
ERR_CANNOTDOCOMMAND, 24
ERR_CANNOTKNOCK, 22
ERR_CANNOTSENDTOCHAN, 20
ERR_CANNOTSENDTONICK, 24
ERR_CANTKICKADMIN, 22
ERR_CANTKILLSERVER, 22
ERR_CHANNELISFULL, 21
ERR_CHANOPEN, 24
ERR_CHANOPRIVSNEEDED, 22
ERR_CHANOWNPRIVNEEDED, 22
ERR_CHANTOORECENT, 22
ERR_DEAD, 21
ERR_DESYNC, 22
ERR_DISABLED, 22
ERR_DONTCHEAT, 22
ERR_ERRONEUSNICKNAME, 20
ERR_EVENTNICKCHANGE, 20
ERR_FILEERROR, 20
ERR_GHOSTEDCLIENT, 22
ERR_HOSTILENAME, 21
ERR_HOSTUNAVAIL, 23
ERR_HTMDISABLED, 22
ERR_IDCOLLISION, 21
ERR_INVALID_ERROR, 22
ERR_INVALIDUSERNAME, 21
ERR_INVITEONLYCHAN, 21
ERR_ISCHANSERVICE, 22
ERR_ISOPERLCHAN, 22
ERR_ISREALSERVICE, 22
ERR_KEYSET, 21
ERR_KICKEDFROMCHAN, 21
ERR_KILLDENY, 22
ERR_KNOCKDISABLED, 24
ERR_KNOCKONCHAN, 24
ERR_LENGTHTRUNCATED, 20
ERR_LINKCHANNEL, 21
ERR_LINKFAIL, 21
ERR_LINKSET, 21
ERR_LISTSYNTAX, 23
ERR_LONGMASK, 22
ERR_MASKTOOWIDE, 23
ERR_MSGSERVICES, 22
ERR_NEEDMOREPARAMS, 21
ERR_NEEDREGGEDNICK, 21
ERR_NICKCOLLISION, 20
ERR_NICKLOST, 21
ERR_NICKNAMEINUSE, 20
ERR_NICKTOOFAST, 21
ERR_NOADMININFO, 20
ERR_NOCHANMODES, 21
ERR_NOCOLORSONCHAN, 20
ERR_NOFEATURE, 22
ERR_NOHIDING, 21
ERR_NOINVITE, 22
ERR_NOLANGUAGE, 24
ERR_NOLOGIN, 21
ERR_NOMOTD, 20
ERR_NONICKCHANGE, 21
ERR_NONICKNAMEGIVEN, 20
ERR_NONONREG, 22
ERR_NOOPERHOST, 22
ERR_NOOPERMOTD, 20
ERR_NOORIGIN, 20
ERR_NOPERMFORHOST, 21
ERR_NOPRIVILEGES, 22
ERR_NOPRIVS, 24
ERR_NORECIPIENT, 20
ERR_NORULES, 20
ERR_NOSERVICEHOST, 22
ERR_NOSUCHCHANNEL, 20
ERR_NOSUCHNICK, 20
ERR_NOSUCHSERVER, 20
ERR_NOSUCHSERVICE, 20
ERR_NOTEXTTOSEND, 20
ERR_NOTFORHALFOPS, 21
ERR_NOTIMPLEMENTED, 21
ERR_NOTONCHANNEL, 21
ERR_NOTOPLEVEL, 20
ERR_NOTREGISTERED, 21
ERR_NOULINE, 22
ERR_NUMERIC_ERR, 24
ERR_ONLYSERVERSCANCHANGE, 21
ERR_OPERONLY, 23
ERR_OPERSPVERIFY, 23
ERR_PASSWDMISMATCH, 21
ERR_PFXUNROUTABLE, 23
ERR_QUARANTINED, 23
ERR_QUERYTOOLONG, 20
ERR_REMOTEPFX, 23

ERR_RESTRICTED, 22
ERR_SECUREONLYCHAN, 22
ERR_SERVERMODELOCK, 24
ERR_SERVICECONFUSED, 20
ERR_SERVICENAMEINUSE, 20
ERR_SERVICESDOWN, 21
ERR_SILELISTFULL, 22
ERR_STATSSLINE, 23
ERR_SUMMONDISABLED, 21
ERR_TARGETTOOFAST, 21
ERR_TEXTTOOSHORT, 24
ERR_TOOMANYAWAY, 20
ERR_TOOMANYCHANNELS, 20
ERR_TOOMANYDCC, 22
ERR_TOOMANYKNOCK, 24
ERR_TOOMANYLANGUAGES, 24
ERR_TOOMANYMATCHES, 20
ERR_TOOMANYTARGETS, 20
ERR_TOOMANYUSERS, 23
ERR_TOOMANYWATCH, 22
ERR_TSLESSCHAN, 22
ERR_UMODEUNKNOWNFLAG, 22
ERR_UNAVAILRESOURCE, 20
ERR_UNIQOPRIVSNEEDED, 22
ERR_UNKNOWNCOMMAND, 20
ERR_UNKNOWNERROR, 20
ERR_UNKNOWNMODE, 21
ERR_UNKNOWNSERVERMODE, 24
ERR_USERNOTINCHANNEL, 21
ERR_USERNOTONSERV, 22
ERR_USERONCHANNEL, 21
ERR_USERSDISABLED, 21
ERR_USERSDONTMATCH, 22
ERR_USINGSLINE, 23
ERR_VOICENEEDED, 22
ERR_VWORLDWARN, 22
ERR_WASNOSUCHNICK, 20
ERR_WHOLIMEXCEED, 23
ERR_WHOSYNTAX, 23
ERR_WHOTRUNC, 23
ERR_WILDTOPLEVEL, 20
ERR_YOUREBANNEDCREEP, 21
ERR_YOUWILLBEBANNED, 21
RPL_ACCEPTLIST, 17
RPL_ADMINEMAIL, 16
RPL_ADMINLOC1, 16
RPL_ADMINLOC2, 16
RPL_ADMINME, 16
RPL_ALIST, 20
RPL_ATTEMPTINGJUNC, 15
RPL_ATTEMPTINGREROUTE, 15
RPL_AWAY, 17
RPL_BADCHANPASS, 19
RPL_BANEXPIRED, 19
RPL_BANLINKED, 19
RPL_BANLIST, 19
RPL_BOUNCE, 15
RPL_CANNOTSETMODES, 24
RPL_CHANINFO_AWAY, 17
RPL_CHANINFO_BANNED, 17
RPL_CHANINFO_BANS, 17
RPL_CHANINFO_CHOPS, 17
RPL_CHANINFO_HANDLE, 17
RPL_CHANINFO_INVITE, 17
RPL_CHANINFO_INVITES, 17
RPL_CHANINFO_KICK, 17
RPL_CHANINFO_KICKS, 17
RPL_CHANINFO_OPERS, 17
RPL_CHANINFO_USERS, 17
RPL_CHANINFO_VOICES, 17
RPL_CHANNEL_URL, 18
RPL_CHANNELMODEIS, 18
RPL_CHANNELPASSIS, 18
RPL_CHANPASSOK, 19
RPL_CHANREDIR, 23
RPL_CHANUSER, 17
RPL_CHKHEAD, 17
RPL_CHPASSUNKNOWN, 18
RPL_CLOSEEND, 19
RPL_CLOSING, 19
RPL_COMMANDSYNTAX, 18
RPL_CREATED, 15
RPL_CREATIONTIME, 18
RPL_DATASTR, 17
RPL_DCCINFO, 23
RPL_DCCLIST, 23
RPL_DCCSTATUS, 23
RPL_DUMPING, 23
RPL_DUMPRPL, 23
RPL_END_CHANINFO, 17
RPL_END_NETSTAT, 16
RPL_ENDNOTIFY, 17
RPL_ENDOF_GENERIC, 24
RPL_ENDOFACCEPT, 17
RPL_ENDOFALIST, 20
RPL_ENDOFBANLIST, 19
RPL_ENDOFCHECK, 17
RPL_ENDOFDCCLIST, 23
RPL_ENDOFEXCEPTLIST, 19
RPL_ENDOFGLIST, 17
RPL_ENDOFHELP, 24
RPL_ENDOFINFO, 19
RPL_ENDOFINVITELIST, 19
RPL_ENDOFIRCOPS, 20
RPL_ENDOFISOPER, 23
RPL_ENDOFJUPELIST, 17
RPL_ENDOFLINKS, 19
RPL_ENDOFMOTD, 19
RPL_ENDOFNAMES, 19
RPL_ENDOFO, 23
RPL_ENDOFOMOTD, 24
RPL_ENDOFQLIST, 20
RPL_ENDOFSERVICES, 16
RPL_ENDOFSETTINGS, 23
RPL_ENDOFSILELIST, 17
RPL_ENDOFSTATS, 16

RPL_ENDOFUSERS, 20
RPL_ENDOFWATCHLIST, 23
RPL_ENDOFWHO, 18
RPL_ENDOFWHOIS, 18
RPL_ENDOFWHOWAS, 19
RPL_EODUMP, 23
RPL_ETRACE, 24
RPL_ETRACEFULL, 24
RPL_EXCEPTLIST, 19
RPL_FEATURE, 17
RPL_GLIST, 17
RPL_GLIST_HASH, 17
RPL_GLOBALUSERS, 16
RPL_HELPFWD, 17
RPL_HELPHDR, 17
RPL_HELPHLP, 17
RPL_HELPIGN, 17
RPL_HELPOP, 17
RPL_HELPSTART, 24
RPL_HELPTLR, 17
RPL_HELPTXT, 24
RPL_HOSTHIDDEN, 20
RPL_INFO, 19
RPL_INFOSTART, 19
RPL_INVITED, 19
RPL_INVITELIST, 19
RPL_INVITING, 19
RPL_IRCOPS, 19
RPL_ISLOCOP, 23
RPL_ISNOTOPER, 23
RPL_ISON, 18
RPL_ISOPER, 23
RPL_ISUPPORT, 15
RPL_JUPELIST, 17
RPL_KICKEXPIRED, 19
RPL_KICKLINKED, 19
RPL_KILLDONE, 19
RPL_KNOCK, 24
RPL_KNOCKDLVR, 24
RPL_LANGUAGE, 24
RPL_LINKS, 19
RPL_LIST, 18
RPL_LISTEND, 18
RPL_LISTSTART, 18
RPL_LISTSYNTAX, 18
RPL_LISTUSAGE, 18
RPL_LOCALUSERS, 16
RPL_LOGOFF, 23
RPL_LOGON, 23
RPL_LUSERCHANNELS, 16
RPL_LUSERCLIENT, 16
RPL_LUSERME, 16
RPL_LUSEROP, 16
RPL_LUSERSTAFF, 24
RPL_LUSERUNKNOWN, 16
RPL_MODECHANGEWARN, 23
RPL_MODLIST, 16
RPL_MOTD, 19
RPL_MOTDSTART, 19
RPL_MYINFO, 15
RPL_MYPORTIS, 19
RPL_NAMREPLY, 19
RPL_NAMREPLY_, 19
RPL_NETSTAT, 16
RPL_NETWORKS, 24
RPL_NEWHOSTIS, 17
RPL_NICKTRACE, 18
RPL_NOCHANPASS, 18
RPL_NONE, 17
RPL_NOTESTLINE, 24
RPL_NOTIFY, 17
RPL_NOTIFYACTION, 18
RPL_NOTOPERANYMORE, 19
RPL_NOTOPIC, 18
RPL_NOUSERS, 20
RPL_NOWAWAY, 18
RPL_NOWOFF, 23
RPL_NOWON, 23
RPL_OTHERUMODEIS, 24
RPL_PATCHCON, 17
RPL_PATCHHEAD, 17
RPL_PRIVS, 17
RPL_QLIST, 19
RPL_REHASHING, 19
RPL_SAVENICK, 15
RPL_SERVICE, 16
RPL_SERVICEINFO, 16
RPL_SERVLIST, 16
RPL_SERVLISTEND, 16
RPL_SERVMODEIS, 23
RPL_SETTINGS, 23
RPL_SILELIST, 17
RPL_SNOMASK, 15
RPL_SPAM, 19
RPL_SQLINE_NICK, 16
RPL_START_NETSTAT, 16
RPL_STATMEM, 15
RPL_STATMEMTOT, 15
RPL_STATS, 15
RPL_STATSBLINE, 16
RPL_STATSCLINE, 15
RPL_STATSCOMMANDS, 15
RPL_STATSCONN, 16
RPL_STATSCOUNT, 16
RPL_STATSDEBUG, 16
RPL_STATSDEFINE, 16
RPL_STATSDELTA, 17
RPL_STATSDLINE, 16
RPL_STATSENGINE, 16
RPL_STATSHLINE, 16
RPL_STATSIAUTH, 16
RPL_STATSILINE, 15
RPL_STATSKLINE, 15
RPL_STATSLINKINFO, 15
RPL_STATSLLINE, 16
RPL_STATSNLINE, 15

RPL_STATSOLINE, 16
RPL_STATSPING, 16
RPL_STATSQLINE, 15
RPL_STATSSLINE, 16
RPL_STATSUPTIME, 16
RPL_STATSVERBOSE, 16
RPL_STATSVLINE, 16
RPL_STATSYLINE, 16
RPL_STATSZLINE, 16
RPL_SUMMONING, 19
RPL_SUSERHOST, 18
RPL_TARGNOTIFY, 24
RPL_TARGUMODEG, 24
RPL_TESTLINE, 24
RPL_TESTMARK, 24
RPL_TEXT, 18
RPL_TIME, 20
RPL_TIMEONSERVERIS, 24
RPL_TOPIC, 18
RPL_TOPICWHOTIME, 18
RPL_TRACECLASS, 15
RPL_TRACECONNECTING, 15
RPL_TRACEEND, 16
RPL_TRACEHANDSHAKE, 15
RPL_TRACELINK, 15
RPL_TRACELOG, 16
RPL_TRACENEWTYPE, 15
RPL_TRACEOPERATOR, 15
RPL_TRACEPING, 16
RPL_TRACERECONNECT, 15
RPL_TRACEROUTE_HOP, 23
RPL_TRACEROUTE_START, 23
RPL_TRACESERVER, 15
RPL_TRACESERVICE, 15
RPL_TRACEUNKNOWN, 15
RPL_TRACEUSER, 15
RPL_TRYAGAIN, 16
RPL_UMODEGMSG, 24
RPL_UMODEIS, 16
RPL_UNAWAY, 18
RPL_UNIQOPIS, 18
RPL_UNKNOWNMODES, 24
RPL_USERHOST, 18
RPL_USERS, 20
RPL_USERSSTART, 20
RPL_VCHANEXIST, 17
RPL_VCHANHELP, 17
RPL_VCHANLIST, 17
RPL_VERSION, 19
RPL_WATCHCLEAR, 23
RPL_WATCHLIST, 23
RPL_WATCHOFF, 23
RPL_WATCHSTAT, 23
RPL_WELCOME, 15
RPL_WHOIS_HIDDEN, 18
RPL_WHOISACCOUNT, 18
RPL_WHOISACTUALLY, 19
RPL_WHOISADMIN, 18
RPL_WHOISCHANNELS, 18
RPL_WHOISCHANOP, 18
RPL_WHOISHELPER, 18
RPL_WHOISHELPOP, 18
RPL_WHOISHOST, 19
RPL_WHOISIDLE, 18
RPL_WHOISLANGUAGE, 24
RPL_WHOISMODES, 19
RPL_WHOISOPERATOR, 18
RPL_WHOISREGNICK, 18
RPL_WHOISSADMIN, 18
RPL_WHOISSECURE, 24
RPL_WHOISSERVER, 18
RPL_WHOISSERVICE, 18
RPL_WHOISSPECIAL, 18
RPL_WHOISSTAFF, 24
RPL_WHOISSVCMSG, 18
RPL_WHOISUSER, 18
RPL_WHOISVIRT, 18
RPL_WHOREPLY, 19
RPL_WHOSPCRPL, 19
RPL_WHOWAS_TIME, 18
RPL_WHOWASDETAILS, 24
RPL_WHOWASHOST, 23
RPL_WHOWASUSER, 18
RPL_XINFO, 24
RPL_XINFOEND, 24
RPL_XINFOSTART, 24
RPL_YOURCOOKIE, 15
RPL_YOUREOPER, 19
RPL_YOURESERVICE, 19
RPL_YOURHELPER, 19
RPL_YOURHOST, 15
RPL_YOURID, 15
RPL_YOURLANGUAGEIS, 24
ircClient
server::ircClient, 151
ircCode
ircCodes, 15
ircCodes, 11
ircCode, 15
ircConnectTimeOut
server::mainApp, 203
ircConnected
server::mainApp, 203
ircDisconnected
server::mainApp, 204
ircId
server::ircUser, 174
ircUser
client::ircUser, 177, 178
server::ircUser, 169
ircUserList
server::ircUserList, 184
ircclient.h
PING_CHECK_INTERVAL, 376
PING_THRESHOLD, 376
PORT, 377

isConnected
client::clsServerConn, 94
isConnecting
client::clsServerConn, 94
isFirst
K4U::clsLog, 76
isStandard
server::ircUser, 171

JSONCOMMAND_CHANGEOWNUSER
K4U, 25
JSONCOMMAND_EVENT
K4U, 25
JSONCOMMAND_GETEVENTS
K4U, 25
JSONCOMMAND_NONE
K4U, 25
JSONCOMMAND_OWNCHAT
K4U, 25
JSONCOMMAND_PING
K4U, 25
JSONCOMMAND_USERINFOREQ
K4U, 25
JSONCOMMAND_USERQUERY
K4U, 25
JsonTokenColon
QtJson, 30
JsonTokenComma
QtJson, 30
JsonTokenCurlyClose
QtJson, 30
JsonTokenCurlyOpen
QtJson, 30
JsonTokenFalse
QtJson, 30
JsonTokenNone
QtJson, 30
JsonTokenNull
QtJson, 30
JsonTokenNumber
QtJson, 30
JsonTokenSquaredClose
QtJson, 30
JsonTokenSquaredOpen
QtJson, 30
JsonTokenString
QtJson, 30
JsonTokenTrue
QtJson, 30
jsonCommand
K4U::jsonCommand, 190, 191
jsonCommands
K4U, 25
JsonToken
QtJson, 30

K4U
JSONCOMMAND_CHANGEOWNUSER, 25
JSONCOMMAND_EVENT, 25
JSONCOMMAND_GETEVENTS, 25
JSONCOMMAND_NONE, 25
JSONCOMMAND_OWNCHAT, 25
JSONCOMMAND_PING, 25
JSONCOMMAND_USERINFOREQ, 25
JSONCOMMAND_USERQUERY, 25
LOGTAGS_EVENT, 26
LOGTAGS_IRC, 26
LOGTAGS_MAIN, 26
LOGTAGS_SERVER, 26
LOGTAGS_UI, 26
LOGTAGS_UNKNOWN, 26
LOGTAGS_USER, 26
LOGTAGS_WEB, 26
K4U, 25
endl, 26
jsonCommands, 25
logTags, 25
operator<<, 27, 28
K4U::clsLog, 69
clsLog, 70
doEndl, 71
getTag, 72
isFirst, 76
operator<<, 72–75
prefix, 75
prf, 76
setPrefix, 75
tag, 76
K4U::jsonCommand, 189
addToData, 191
command, 196
data, 196
getCommand, 192
getCommandString, 192
getData, 193
jsonCommand, 190, 191
operator<<, 196
operator=, 194
setCommand, 194
setCommandInt, 194
setData, 194
toJsonString, 195
keyPressEvent
client::chatInput, 55

LOGTAGS_EVENT
K4U, 26
LOGTAGS_IRC
K4U, 26
LOGTAGS_MAIN
K4U, 26
LOGTAGS_SERVER
K4U, 26
LOGTAGS_UI
K4U, 26
LOGTAGS_UNKNOWN
K4U, 26
LOGTAGS_USER

K4U, 26
LOGTAGS_WEB
K4U, 26
LIB_PATH_KEY
org::kde::necessitas::origo::QtActivity, 258
lastId
server::ircUserList, 189
lastPing
server::ircClient, 165
level
shared, 36
listeningPort
server::clsSettings, 106
loadApplication
org::kde::necessitas::origo::QtActivity, 238
loaderReady
org::kde::necessitas::ministro::IMinistroCallback, 145
log
client::clsServerConn, 95
client::ircUser, 182
client::MainWindow, 229
server::ircClient, 165
server::ircUser, 174
server::mainApp, 211
server::uiClient, 276
server::uiServer, 287
shared::events::clsEvent, 69
logTags
K4U, 25
lostConnection
server::uiServer, 283

m_activityInfo
org::kde::necessitas::origo::QtActivity, 259
m_classLoader
org::kde::necessitas::origo::QtActivity, 259
m_delegateMethods
org::kde::necessitas::origo::QtApplication, 266
m_delegateObject
org::kde::necessitas::origo::QtApplication, 267
m_ministroConnection
org::kde::necessitas::origo::QtActivity, 259
m_qtLibs
org::kde::necessitas::origo::QtActivity, 259
MAIN_LIBRARY_KEY
org::kde::necessitas::origo::QtActivity, 259
MAX_EVENTS
mainapp.h, 391
main
customIrcClient/android/main.cpp, 295
customIrcClient/main.cpp, 296
customIrcServer/main.cpp, 296
MainWindow
ScreenOrientationAuto, 214
ScreenOrientationLockLandscape, 214
ScreenOrientationLockPortrait, 214
mainApp
server::mainApp, 200
MainWindow, 212
~MainWindow, 214
client::MainWindow, 218
MainWindow, 214
MainWindow, 214
ScreenOrientation, 214
setOrientation, 214
showExpanded, 215
ui, 215
mainapp.h
MAX_EVENTS, 391
RECONNECT_IN_S, 391
mainwindow.h
RECONNECT_IN_S, 305
message
client::chatEntry, 52
client::chatNotification, 62
ministroNotFound
org::kde::necessitas::origo::QtActivity, 238
motdReceived
server::ircClient, 165
moveScrollBarToBottom
client::MainWindow, 222

name
client::chatEntry, 52
server::ircUser, 174
nick
client::ircUser, 182
server::nickAndStatus, 230

onActivityResult
org::kde::necessitas::origo::QtActivity, 239
org::kde::necessitas::origo::QtApplication, 267
onApplyThemeResource
org::kde::necessitas::origo::QtActivity, 239
onAttachedToWindow
org::kde::necessitas::origo::QtActivity, 239
onBackPressed
org::kde::necessitas::origo::QtActivity, 239
onChildTitleChanged
org::kde::necessitas::origo::QtActivity, 240
onConfigurationChanged
org::kde::necessitas::origo::QtActivity, 240
onContentChanged
org::kde::necessitas::origo::QtActivity, 240
onContextItemSelected
org::kde::necessitas::origo::QtActivity, 240
onContextMenuClosed
org::kde::necessitas::origo::QtActivity, 241
onCreate
org::kde::necessitas::origo::QtActivity, 241
org::kde::necessitas::origo::QtApplication, 267
onCreateContextMenu
org::kde::necessitas::origo::QtActivity, 241
onCreateDescription
org::kde::necessitas::origo::QtActivity, 241
onCreateDialog
org::kde::necessitas::origo::QtActivity, 242

onCreateOptionsMenu
org::kde::necessitas::origo::QtActivity, 242
onCreatePanelMenu
org::kde::necessitas::origo::QtActivity, 242
onCreatePanelView
org::kde::necessitas::origo::QtActivity, 243
onCreateThumbnail
org::kde::necessitas::origo::QtActivity, 243
onCreateView
org::kde::necessitas::origo::QtActivity, 243
onDestroy
org::kde::necessitas::origo::QtActivity, 243
onDetachedFromWindow
org::kde::necessitas::origo::QtActivity, 244
onGenericMotionEvent
org::kde::necessitas::origo::QtApplication, 267
onKeyDown
org::kde::necessitas::origo::QtActivity, 244
org::kde::necessitas::origo::QtApplication, 267
onKeyLongPress
org::kde::necessitas::origo::QtActivity, 244
org::kde::necessitas::origo::QtApplication, 267
onKeyMultiple
org::kde::necessitas::origo::QtActivity, 245
org::kde::necessitas::origo::QtApplication, 267
onKeyShortcut
org::kde::necessitas::origo::QtApplication, 267
onKeyUp
org::kde::necessitas::origo::QtActivity, 245
org::kde::necessitas::origo::QtApplication, 267
onLowMemory
org::kde::necessitas::origo::QtActivity, 245
onMenuItemSelected
org::kde::necessitas::origo::QtActivity, 246
onMenuOpened
org::kde::necessitas::origo::QtActivity, 246
onNewIntent
org::kde::necessitas::origo::QtActivity, 246
onOptionsItemSelected
org::kde::necessitas::origo::QtActivity, 246
onOptionsMenuClosed
org::kde::necessitas::origo::QtActivity, 247
onPanelClosed
org::kde::necessitas::origo::QtActivity, 247
onPause
org::kde::necessitas::origo::QtActivity, 247
onPostCreate
org::kde::necessitas::origo::QtActivity, 247
onPostResume
org::kde::necessitas::origo::QtActivity, 248
onPrepareDialog
org::kde::necessitas::origo::QtActivity, 248
onPrepareOptionsMenu
org::kde::necessitas::origo::QtActivity, 248
onPreparePanel
org::kde::necessitas::origo::QtActivity, 249
onRestart
org::kde::necessitas::origo::QtActivity, 249
onRestoreInstanceState
org::kde::necessitas::origo::QtActivity, 249
onResume
org::kde::necessitas::origo::QtActivity, 249
onRetainNonConfigurationInstance
org::kde::necessitas::origo::QtActivity, 250
onSaveInstanceState
org::kde::necessitas::origo::QtActivity, 250
onSearchRequested
org::kde::necessitas::origo::QtActivity, 250
onStart
org::kde::necessitas::origo::QtActivity, 250
onStop
org::kde::necessitas::origo::QtActivity, 251
onTerminate
org::kde::necessitas::origo::QtApplication, 265
onTitleChanged
org::kde::necessitas::origo::QtActivity, 251
onTouchEvent
org::kde::necessitas::origo::QtActivity, 251
org::kde::necessitas::origo::QtApplication, 267
onTrackballEvent
org::kde::necessitas::origo::QtActivity, 251
org::kde::necessitas::origo::QtApplication, 267
onUserInteraction
org::kde::necessitas::origo::QtActivity, 252
onUserLeaveHint
org::kde::necessitas::origo::QtActivity, 252
onWindowAttributesChanged
org::kde::necessitas::origo::QtActivity, 252
onWindowFocusChanged
org::kde::necessitas::origo::QtActivity, 252
online
client::ircUser, 182
server::ircUser, 174
operator<<
K4U, 27, 28
K4U::clsLog, 72–75
K4U::jsonCommand, 196
server, 35
server::ircUser, 174
operator=
K4U::jsonCommand, 194
server::ircUser, 171
org, 28
org.k4u, 28
org.k4u.customIrcClient, 28
org.k4u.customIrcClient.BuildConfig, 39
org.k4u.customIrcClient.R, 268
org.kde, 29
org.kde.necessitas, 29
org.kde.necessitas.ministro, 29
org.kde.necessitas.ministro.IMinistro, 142
org.kde.necessitas.ministro.IMinistroCallback, 144
org.kde.necessitas.origo, 29
org.kde.necessitas.origo.QtActivity, 232
org.kde.necessitas.origo.QtApplication, 261
org::k4u::customIrcClient::BuildConfig

DEBUG, 39
org::kde::necessitas::ministro::IMinistro
requestLoader, 143
org::kde::necessitas::ministro::IMinistroCallback
loaderReady, 145
org::kde::necessitas::origo::QtActivity
DEX_PATH_KEY, 258
dispatchKeyEvent, 236
dispatchPopulateAccessibilityEvent, 236
dispatchTouchEvent, 237
dispatchTrackballEvent, 237
downloadUpgradeMinistro, 237
LIB_PATH_KEY, 258
loadApplication, 238
m_activityInfo, 259
m_classLoader, 259
m_ministroConnection, 259
m_qtLibs, 259
ministroNotFound, 238
onActivityResult, 239
onApplyThemeResource, 239
onAttachedToWindow, 239
onBackPressed, 239
onChildTitleChanged, 240
onConfigurationChanged, 240
onContentChanged, 240
onContextItemSelected, 240
onContextMenuClosed, 241
onCreate, 241
onCreateContextMenu, 241
onCreateDescription, 241
onCreateDialog, 242
onCreateOptionsMenu, 242
onCreatePanelMenu, 242
onCreatePanelView, 243
onCreateThumbnail, 243
onCreateView, 243
onDestroy, 243
onDetachedFromWindow, 244
onKeyDown, 244
onKeyLongPress, 244
onKeyMultiple, 245
onKeyUp, 245
onLowMemory, 245
onMenuItemSelected, 246
onMenuOpened, 246
onNewIntent, 246
onOptionsItemSelected, 246
onOptionsMenuClosed, 247
onPanelClosed, 247
onPause, 247
onPostCreate, 247
onPostResume, 248
onPrepareDialog, 248
onPrepareOptionsMenu, 248
onPreparePanel, 249
onRestart, 249
onRestoreInstanceState, 249
onResume, 249
onRetainNonConfigurationInstance, 250
onSaveInstanceState, 250
onSearchRequested, 250
onStart, 250
onStop, 251
onTitleChanged, 251
onTouchEvent, 251
onTrackballEvent, 251
onUserInteraction, 252
onUserLeaveHint, 252
onWindowAttributesChanged, 252
onWindowFocusChanged, 252
QT_PROVIDER, 260
QT_VERSION, 260
startApp, 253
super_dispatchKeyEvent, 253
super_dispatchPopulateAccessibilityEvent, 253
super_dispatchTouchEvent, 253
super_dispatchTrackballEvent, 254
super_onActivityResult, 254
super_onApplyThemeResource, 254
super_onAttachedToWindow, 254
super_onBackPressed, 254
super_onChildTitleChanged, 254
super_onConfigurationChanged, 254
super_onContentChanged, 254
super_onContextItemSelected, 254
super_onContextMenuClosed, 254
super_onCreateContextMenu, 254
super_onCreateDescription, 255
super_onCreateDialog, 255
super_onCreateOptionsMenu, 255
super_onCreatePanelMenu, 255
super_onCreatePanelView, 255
super_onCreateThumbnail, 255
super_onCreateView, 255
super_onDetachedFromWindow, 255
super_onKeyDown, 255
super_onKeyLongPress, 255
super_onKeyMultiple, 255
super_onKeyUp, 256
super_onMenuItemSelected, 256
super_onMenuOpened, 256
super_onNewIntent, 256
super_onOptionsItemSelected, 256
super_onOptionsMenuClosed, 256
super_onPanelClosed, 256
super_onPrepareDialog, 256
super_onPrepareOptionsMenu, 256
super_onPreparePanel, 256
super_onRestoreInstanceState, 256
super_onRetainNonConfigurationInstance, 257
super_onSaveInstanceState, 257
super_onSearchRequested, 257
super_onTitleChanged, 257
super_onTouchEvent, 257
super_onTrackballEvent, 257

super_onUserInteraction, 257
super_onUserLeaveHint, 257
super_onWindowAttributesChanged, 257
super_onWindowFocusChanged, 257
org::kde::necessitas::origo::QtApplication
dispatchGenericMotionEvent, 266
dispatchKeyEvent, 266
dispatchKeyShortcutEvent, 266
dispatchPopulateAccessibilityEvent, 266
dispatchTouchEvent, 266
dispatchTrackballEvent, 266
invokeDelegate, 263
invokeDelegateMethod, 264
m_delegateMethods, 266
m_delegateObject, 267
onActivityResult, 267
onCreate, 267
onGenericMotionEvent, 267
onKeyDown, 267
onKeyLongPress, 267
onKeyMultiple, 267
onKeyShortcut, 267
onKeyUp, 267
onTerminate, 265
onTouchEvent, 267
onTrackballEvent, 267
QtTAG, 267
setQtActivityDelegate, 266
stackDeep, 268
ourType
shared::events::clsEvent, 69
ownNickAndStatus
server::ircClient, 165
ownUser
server::clsSettings, 106
server::ircClient, 165
server::mainApp, 211
ownUserChangeStatus
server::mainApp, 204

PING_CHECK_INTERVAL
ircclient.h, 376
PING_THRESHOLD
ircclient.h, 376
PORT
ircclient.h, 377
parse
QtJson, 30
port
client::clsServerConn, 95
client::clsSettings, 97
server::ircClient, 165
server::uiServer, 287
prefix
K4U::clsLog, 75
prf
K4U::clsLog, 76

QT_PROVIDER
org::kde::necessitas::origo::QtActivity, 260
QT_PROVIDER_KEY
org::kde::necessitas::origo::QtActivity, 260
QT_VERSION
org::kde::necessitas::origo::QtActivity, 260
QtJson
JsonTokenColon, 30
JsonTokenComma, 30
JsonTokenCurlyClose, 30
JsonTokenCurlyOpen, 30
JsonTokenFalse, 30
JsonTokenNone, 30
JsonTokenNull, 30
JsonTokenNumber, 30
JsonTokenSquaredClose, 30
JsonTokenSquaredOpen, 30
JsonTokenString, 30
JsonTokenTrue, 30
QtJson, 29
JsonToken, 30
parse, 30
serialize, 31
QtTAG
org::kde::necessitas::origo::QtApplication, 267
qts
server, 35

r
server::ircUser::userColor, 288
RPL_ACCEPTLIST
ircCodes, 17
RPL_ADMINEMAIL
ircCodes, 16
RPL_ADMINLOC1
ircCodes, 16
RPL_ADMINLOC2
ircCodes, 16
RPL_ADMINME
ircCodes, 16
RPL_ALIST
ircCodes, 20
RPL_ATTEMPTINGJUNC
ircCodes, 15
RPL_ATTEMPTINGREROUTE
ircCodes, 15
RPL_AWAY
ircCodes, 17
RPL_BADCHANPASS
ircCodes, 19
RPL_BANEXPIRED
ircCodes, 19
RPL_BANLINKED
ircCodes, 19
RPL_BANLIST
ircCodes, 19
RPL_BOUNCE
ircCodes, 15
RPL_CANNOTSETMODES
ircCodes, 24

RPL_CHANINFO_AWAY
ircCodes, 17
RPL_CHANINFO_BANNED
ircCodes, 17
RPL_CHANINFO_BANS
ircCodes, 17
RPL_CHANINFO_CHOPS
ircCodes, 17
RPL_CHANINFO_HANDLE
ircCodes, 17
RPL_CHANINFO_INVITE
ircCodes, 17
RPL_CHANINFO_INVITES
ircCodes, 17
RPL_CHANINFO_KICK
ircCodes, 17
RPL_CHANINFO_KICKS
ircCodes, 17
RPL_CHANINFO_OPERS
ircCodes, 17
RPL_CHANINFO_USERS
ircCodes, 17
RPL_CHANINFO_VOICES
ircCodes, 17
RPL_CHANNEL_URL
ircCodes, 18
RPL_CHANNELMODEIS
ircCodes, 18
RPL_CHANNELPASSIS
ircCodes, 18
RPL_CHANPASSOK
ircCodes, 19
RPL_CHANREDIR
ircCodes, 23
RPL_CHANUSER
ircCodes, 17
RPL_CHKHEAD
ircCodes, 17
RPL_CHPASSUNKNOWN
ircCodes, 18
RPL_CLOSEEND
ircCodes, 19
RPL_CLOSING
ircCodes, 19
RPL_COMMANDSYNTAX
ircCodes, 18
RPL_CREATED
ircCodes, 15
RPL_CREATIONTIME
ircCodes, 18
RPL_DATASTR
ircCodes, 17
RPL_DCCINFO
ircCodes, 23
RPL_DCCLIST
ircCodes, 23
RPL_DCCSTATUS
ircCodes, 23
RPL_DUMPING
ircCodes, 23
RPL_DUMPRPL
ircCodes, 23
RPL_END_CHANINFO
ircCodes, 17
RPL_END_NETSTAT
ircCodes, 16
RPL_ENDNOTIFY
ircCodes, 17
RPL_ENDOF_GENERIC
ircCodes, 24
RPL_ENDOFACCEPT
ircCodes, 17
RPL_ENDOFALIST
ircCodes, 20
RPL_ENDOFBANLIST
ircCodes, 19
RPL_ENDOFCHECK
ircCodes, 17
RPL_ENDOFDCCLIST
ircCodes, 23
RPL_ENDOFEXCEPTLIST
ircCodes, 19
RPL_ENDOFGLIST
ircCodes, 17
RPL_ENDOFHELP
ircCodes, 24
RPL_ENDOFINFO
ircCodes, 19
RPL_ENDOFINVITELIST
ircCodes, 19
RPL_ENDOFIRCOPS
ircCodes, 20
RPL_ENDOFISOPER
ircCodes, 23
RPL_ENDOFJUPELIST
ircCodes, 17
RPL_ENDOFLINKS
ircCodes, 19
RPL_ENDOFMOTD
ircCodes, 19
RPL_ENDOFNAMES
ircCodes, 19
RPL_ENDOFO
ircCodes, 23
RPL_ENDOFOMOTD
ircCodes, 24
RPL_ENDOFQLIST
ircCodes, 20
RPL_ENDOFSERVICES
ircCodes, 16
RPL_ENDOFSETTINGS
ircCodes, 23
RPL_ENDOFSILELIST
ircCodes, 17
RPL_ENDOFSTATS
ircCodes, 16

RPL_ENDOFUSERS
ircCodes, 20
RPL_ENDOFWATCHLIST
ircCodes, 23
RPL_ENDOFWHO
ircCodes, 18
RPL_ENDOFWHOIS
ircCodes, 18
RPL_ENDOFWHOWAS
ircCodes, 19
RPL_EODUMP
ircCodes, 23
RPL_ETRACE
ircCodes, 24
RPL_ETRACEFULL
ircCodes, 24
RPL_EXCEPTLIST
ircCodes, 19
RPL_FEATURE
ircCodes, 17
RPL_GLIST
ircCodes, 17
RPL_GLIST_HASH
ircCodes, 17
RPL_GLOBALUSERS
ircCodes, 16
RPL_HELPFWD
ircCodes, 17
RPL_HELPHDR
ircCodes, 17
RPL_HELPHLP
ircCodes, 17
RPL_HELPIGN
ircCodes, 17
RPL_HELPOP
ircCodes, 17
RPL_HELPSTART
ircCodes, 24
RPL_HELPTLR
ircCodes, 17
RPL_HELPTXT
ircCodes, 24
RPL_HOSTHIDDEN
ircCodes, 20
RPL_INFO
ircCodes, 19
RPL_INFOSTART
ircCodes, 19
RPL_INVITED
ircCodes, 19
RPL_INVITELIST
ircCodes, 19
RPL_INVITING
ircCodes, 19
RPL_IRCOPS
ircCodes, 19
RPL_ISLOCOP
ircCodes, 23
RPL_ISNOTOPER
ircCodes, 23
RPL_ISON
ircCodes, 18
RPL_ISOPER
ircCodes, 23
RPL_ISUPPORT
ircCodes, 15
RPL_JUPELIST
ircCodes, 17
RPL_KICKEXPIRED
ircCodes, 19
RPL_KICKLINKED
ircCodes, 19
RPL_KILLDONE
ircCodes, 19
RPL_KNOCK
ircCodes, 24
RPL_KNOCKDLVR
ircCodes, 24
RPL_LANGUAGE
ircCodes, 24
RPL_LINKS
ircCodes, 19
RPL_LIST
ircCodes, 18
RPL_LISTEND
ircCodes, 18
RPL_LISTSTART
ircCodes, 18
RPL_LISTSYNTAX
ircCodes, 18
RPL_LISTUSAGE
ircCodes, 18
RPL_LOCALUSERS
ircCodes, 16
RPL_LOGOFF
ircCodes, 23
RPL_LOGON
ircCodes, 23
RPL_LUSERCHANNELS
ircCodes, 16
RPL_LUSERCLIENT
ircCodes, 16
RPL_LUSERME
ircCodes, 16
RPL_LUSEROP
ircCodes, 16
RPL_LUSERSTAFF
ircCodes, 24
RPL_LUSERUNKNOWN
ircCodes, 16
RPL_MODECHANGEWARN
ircCodes, 23
RPL_MODLIST
ircCodes, 16
RPL_MOTD
ircCodes, 19

RPL_MOTDSTART
ircCodes, 19
RPL_MYINFO
ircCodes, 15
RPL_MYPORTIS
ircCodes, 19
RPL_NAMREPLY
ircCodes, 19
RPL_NAMREPLY_
ircCodes, 19
RPL_NETSTAT
ircCodes, 16
RPL_NETWORKS
ircCodes, 24
RPL_NEWHOSTIS
ircCodes, 17
RPL_NICKTRACE
ircCodes, 18
RPL_NOCHANPASS
ircCodes, 18
RPL_NONE
ircCodes, 17
RPL_NOTESTLINE
ircCodes, 24
RPL_NOTIFY
ircCodes, 17
RPL_NOTIFYACTION
ircCodes, 18
RPL_NOTOPERANYMORE
ircCodes, 19
RPL_NOTOPIC
ircCodes, 18
RPL_NOUSERS
ircCodes, 20
RPL_NOWAWAY
ircCodes, 18
RPL_NOWOFF
ircCodes, 23
RPL_NOWON
ircCodes, 23
RPL_OTHERUMODEIS
ircCodes, 24
RPL_PATCHCON
ircCodes, 17
RPL_PATCHHEAD
ircCodes, 17
RPL_PRIVS
ircCodes, 17
RPL_QLIST
ircCodes, 19
RPL_REHASHING
ircCodes, 19
RPL_SAVENICK
ircCodes, 15
RPL_SERVICE
ircCodes, 16
RPL_SERVICEINFO
ircCodes, 16
RPL_SERVLIST
ircCodes, 16
RPL_SERVLISTEND
ircCodes, 16
RPL_SERVMODEIS
ircCodes, 23
RPL_SETTINGS
ircCodes, 23
RPL_SILELIST
ircCodes, 17
RPL_SNOMASK
ircCodes, 15
RPL_SPAM
ircCodes, 19
RPL_SQLINE_NICK
ircCodes, 16
RPL_START_NETSTAT
ircCodes, 16
RPL_STATMEM
ircCodes, 15
RPL_STATMEMTOT
ircCodes, 15
RPL_STATS
ircCodes, 15
RPL_STATSBLINE
ircCodes, 16
RPL_STATSCLINE
ircCodes, 15
RPL_STATSCOMMANDS
ircCodes, 15
RPL_STATSCONN
ircCodes, 16
RPL_STATSCOUNT
ircCodes, 16
RPL_STATSDEBUG
ircCodes, 16
RPL_STATSDEFINE
ircCodes, 16
RPL_STATSDELTA
ircCodes, 17
RPL_STATSDLINE
ircCodes, 16
RPL_STATSENGINE
ircCodes, 16
RPL_STATSHLINE
ircCodes, 16
RPL_STATSIAUTH
ircCodes, 16
RPL_STATSILINE
ircCodes, 15
RPL_STATSKLINE
ircCodes, 15
RPL_STATSLINKINFO
ircCodes, 15
RPL_STATSLLINE
ircCodes, 16
RPL_STATSNLINE
ircCodes, 15

RPL_STATSOLINE
ircCodes, 16
RPL_STATSPING
ircCodes, 16
RPL_STATSQLINE
ircCodes, 15
RPL_STATSSLINE
ircCodes, 16
RPL_STATSUPTIME
ircCodes, 16
RPL_STATSVERBOSE
ircCodes, 16
RPL_STATSVLINE
ircCodes, 16
RPL_STATSYLINE
ircCodes, 16
RPL_STATSZLINE
ircCodes, 16
RPL_SUMMONING
ircCodes, 19
RPL_SUSERHOST
ircCodes, 18
RPL_TARGNOTIFY
ircCodes, 24
RPL_TARGUMODEG
ircCodes, 24
RPL_TESTLINE
ircCodes, 24
RPL_TESTMARK
ircCodes, 24
RPL_TEXT
ircCodes, 18
RPL_TIME
ircCodes, 20
RPL_TIMEONSERVERIS
ircCodes, 24
RPL_TOPIC
ircCodes, 18
RPL_TOPICWHOTIME
ircCodes, 18
RPL_TRACECLASS
ircCodes, 15
RPL_TRACECONNECTING
ircCodes, 15
RPL_TRACEEND
ircCodes, 16
RPL_TRACEHANDSHAKE
ircCodes, 15
RPL_TRACELINK
ircCodes, 15
RPL_TRACELOG
ircCodes, 16
RPL_TRACENEWTYPE
ircCodes, 15
RPL_TRACEOPERATOR
ircCodes, 15
RPL_TRACEPING
ircCodes, 16
RPL_TRACERECONNECT
ircCodes, 15
RPL_TRACEROUTE_HOP
ircCodes, 23
RPL_TRACEROUTE_START
ircCodes, 23
RPL_TRACESERVER
ircCodes, 15
RPL_TRACESERVICE
ircCodes, 15
RPL_TRACEUNKNOWN
ircCodes, 15
RPL_TRACEUSER
ircCodes, 15
RPL_TRYAGAIN
ircCodes, 16
RPL_UMODEGMSG
ircCodes, 24
RPL_UMODEIS
ircCodes, 16
RPL_UNAWAY
ircCodes, 18
RPL_UNIQOPIS
ircCodes, 18
RPL_UNKNOWNMODES
ircCodes, 24
RPL_USERHOST
ircCodes, 18
RPL_USERS
ircCodes, 20
RPL_USERSSTART
ircCodes, 20
RPL_VCHANEXIST
ircCodes, 17
RPL_VCHANHELP
ircCodes, 17
RPL_VCHANLIST
ircCodes, 17
RPL_VERSION
ircCodes, 19
RPL_WATCHCLEAR
ircCodes, 23
RPL_WATCHLIST
ircCodes, 23
RPL_WATCHOFF
ircCodes, 23
RPL_WATCHSTAT
ircCodes, 23
RPL_WELCOME
ircCodes, 15
RPL_WHOIS_HIDDEN
ircCodes, 18
RPL_WHOISACCOUNT
ircCodes, 18
RPL_WHOISACTUALLY
ircCodes, 19
RPL_WHOISADMIN
ircCodes, 18

RPL_WHOISCHANNELS
ircCodes, 18
RPL_WHOISCHANOP
ircCodes, 18
RPL_WHOISHELPER
ircCodes, 18
RPL_WHOISHELPOP
ircCodes, 18
RPL_WHOISHOST
ircCodes, 19
RPL_WHOISIDLE
ircCodes, 18
RPL_WHOISLANGUAGE
ircCodes, 24
RPL_WHOISMODES
ircCodes, 19
RPL_WHOISOPERATOR
ircCodes, 18
RPL_WHOISREGNICK
ircCodes, 18
RPL_WHOISSADMIN
ircCodes, 18
RPL_WHOISSECURE
ircCodes, 24
RPL_WHOISSERVER
ircCodes, 18
RPL_WHOISSERVICE
ircCodes, 18
RPL_WHOISSPECIAL
ircCodes, 18
RPL_WHOISSTAFF
ircCodes, 24
RPL_WHOISSVCMSG
ircCodes, 18
RPL_WHOISUSER
ircCodes, 18
RPL_WHOISVIRT
ircCodes, 18
RPL_WHOREPLY
ircCodes, 19
RPL_WHOSPCRPL
ircCodes, 19
RPL_WHOWAS_TIME
ircCodes, 18
RPL_WHOWASDETAILS
ircCodes, 24
RPL_WHOWASHOST
ircCodes, 23
RPL_WHOWASUSER
ircCodes, 18
RPL_XINFO
ircCodes, 24
RPL_XINFOEND
ircCodes, 24
RPL_XINFOSTART
ircCodes, 24
RPL_YOURCOOKIE
ircCodes, 15
RPL_YOUREOPER
ircCodes, 19
RPL_YOURESERVICE
ircCodes, 19
RPL_YOURHELPER
ircCodes, 19
RPL_YOURHOST
ircCodes, 15
RPL_YOURID
ircCodes, 15
RPL_YOURLANGUAGEIS
ircCodes, 24
RECONNECT_IN_S
mainapp.h, 391
mainwindow.h, 305
readData
client::clsServerConn, 89
server::ircClient, 157
readServer
client::clsSettings, 97
server::clsSettings, 104
readUiSettings
server::clsSettings, 104
readUserInfo
server::clsSettings, 105
readUsers
server::clsSettings, 105
receiveMessage
server::uiClient, 274
reconnectTimer
client::MainWindow, 229
server::mainApp, 212
refresh
client::clsUserEntry, 110
refreshUser
client::clsUserList, 114
removeConnection
server::uiClient, 275
removeUser
client::clsUserList, 114
requestLoader
org::kde::necessitas::ministro::IMinistro, 143
resizeEvent
client::chatEntry, 51
client::chatNotification, 61
run
server::mainApp, 205

SERVERMESSAGE_LEVEL_ERROR
shared, 36
SERVERMESSAGE_LEVEL_INFO
shared, 36
ScreenOrientationAuto
MainWindow, 214
ScreenOrientationLockLandscape
MainWindow, 214
ScreenOrientationLockPortrait
MainWindow, 214
ScreenOrientation

MainWindow, 214
send
server::ircClient, 158
server::uiClient, 275
server::uiServer, 283, 284
sendChat
client::chatInput, 55
client::MainWindow, 222
server::ircClient, 159, 160
sendCommand
client::clsServerConn, 90
sendPong
server::ircClient, 161
serialize
QtJson, 31
server, 32
client::clsSettings, 98
client::MainWindow, 229
getColor, 33
getId, 33
getNickAndStatus, 33
getNickAndStatusFromId, 34
inCommand, 34
operator<<, 35
qts, 35
server::clsSettings, 106
server::clsSettings, 98
channel, 106
clsSettings, 100
getChannel, 101
getListeningPort, 101
getOwnUser, 102
getServer, 102
getUserName, 103
getUsers, 103
listeningPort, 106
ownUser, 106
readServer, 104
readUiSettings, 104
readUserInfo, 105
readUsers, 105
server, 106
users, 106
server::ircClient, 146
~ircClient, 151
buffer, 164
channel, 164
chatReceived, 151
checkForPing, 152
connectTimeout, 152
connectTimer, 164
connected, 164
connecting, 164
disconnected, 153
doConnect, 153
getOwnNick, 154
handleCommand, 154
hostname, 165
IRCConnectTimeOut, 157
IRCConnected, 156
IRCDisconnected, 157
init, 155
ircClient, 151
lastPing, 165
log, 165
motdReceived, 165
ownNickAndStatus, 165
ownUser, 165
port, 165
readData, 157
send, 158
sendChat, 159, 160
sendPong, 161
setOwnNick, 161
setOwnUser, 161
setUser, 162
sltConnected, 162
sock, 165
startupComplete, 162
suppressDisconnectSignal, 165
timeBetweenPings, 165
timeOutChecker, 165
userChangeNick, 162
userChangeStatus, 163
userOffline, 163
userOnline, 164
server::ircUser, 166
email, 174
getId, 169
getIrcId, 170
getName, 170
getOnline, 170
getStatus, 171
id, 174
ircId, 174
ircUser, 169
isStandard, 171
log, 174
name, 174
online, 174
operator<<, 174
operator=, 171
setColor, 171
setEmail, 171
setId, 172
setIrcId, 172
setNick, 172
setOnline, 172
setStatus, 173
standard, 174
status, 174
toVariantMap, 173
uColor, 174
server::ircUser::userColor, 287
b, 288
g, 288

r, 288
server::ircUserList, 183
add, 184
del, 185
findUser, 186
getAll, 186
getUser, 186
getUserById, 187
getUserByIrcId, 187
getUserByNick, 188
ircUserList, 184
lastId, 189
users, 189
server::mainApp, 196
addToEventHistory, 200
chatReceived, 200
doIrcReconnect, 201
doSendEvents, 202
doUserQuery, 202
events, 211
irc, 211
ircConnectTimeOut, 203
ircConnected, 203
ircDisconnected, 204
log, 211
mainApp, 200
ownUser, 211
ownUserChangeStatus, 204
reconnectTimer, 212
run, 205
serverMessage, 206
settings, 212
TAG, 212
ui, 212
uiSendChat, 207
userChangeNick, 207
userChangeStatus, 208
userOffline, 209
userOnline, 210
users, 212
server::nickAndStatus, 230
nick, 230
status, 231
server::uiClient, 268
buffer, 276
changeOwnUser, 272
chatReceived, 272
checkForJsonCommand, 273
depth, 276
disconnected, 273
getEvents, 274
log, 276
receiveMessage, 274
removeConnection, 275
send, 275
sock, 276
uiClient, 271
userQuery, 276
server::uiServer, 277
~uiServer, 280
acceptConnection, 280
changeOwnUser, 280
chatReceived, 281
connections, 286
doSendEvents, 282
doUserQuery, 282
log, 287
lostConnection, 283
port, 287
send, 283, 284
sgnChangeOwnUserStatus, 285
sgnChatReceived, 285
sgnSendEvents, 285
sgnUserQuery, 286
startListen, 286
uiServer, 279
serverConnected
client::MainWindow, 223
serverDisconnected
client::MainWindow, 224
serverMessage
server::mainApp, 206
serverMessageReceived
client::clsServerConn, 91
client::MainWindow, 224
setAllUserNameSize
client::chatBox, 45
setColor
client::ircUser, 181
server::ircUser, 171
setCommand
K4U::jsonCommand, 194
setCommandInt
K4U::jsonCommand, 194
setData
K4U::jsonCommand, 194
setEmail
client::ircUser, 181
server::ircUser, 171
setEventType
shared::events::clsEvent, 67
setId
server::ircUser, 172
setIrcId
server::ircUser, 172
setNick
client::ircUser, 181
server::ircUser, 172
setOnline
client::ircUser, 181
server::ircUser, 172
setOrientation
MainWindow, 214
setOwnNick
server::ircClient, 161
setOwnUser

server::ircClient, 161
setPrefix
K4U::clsLog, 75
setQtActivityDelegate
org::kde::necessitas::origo::QtApplication, 266
setStatus
client::ircUser, 181
server::ircUser, 173
setUser
server::ircClient, 162
setUserWidth
client::chatEntry, 51
settings
client::MainWindow, 229
server::mainApp, 212
setupUi
client::clsUserList, 114
sgnChangeOwnUserStatus
server::uiServer, 285
sgnChatReceived
server::uiServer, 285
sgnConnectTimeout
client::clsServerConn, 91
sgnSendEvents
server::uiServer, 285
sgnUserQuery
server::uiServer, 286
shared, 36
level, 36
SERVERMESSAGE_LEVEL_ERROR, 36
SERVERMESSAGE_LEVEL_INFO, 36
shared::events
EVENTTYPE_CHAT, 37
EVENTTYPE_NONE, 37
EVENTTYPE_SERVERMSG, 37
EVENTTYPE_USER_JOIN, 37
EVENTTYPE_USER_LEFT, 37
EVENTTYPE_USER_NICK, 37
EVENTTYPE_USER_STATUS, 37
shared::events, 36
eventTypes, 37
shared::events::clsEvent, 62
addToData, 65
clsEvent, 64
data, 68
getData, 65
getEventType, 66
getTime, 67
getType, 67
log, 69
ourType, 69
setEventType, 67
timeOfEvent, 69
toVariant, 68
shared::events::eventChat, 116
eventChat, 118
getMessage, 118
getUser, 119
shared::events::eventServerMessage, 120
eventServerMessage, 122
getLevel, 122
getMessage, 123
shared::events::eventUserChangeNick, 124
eventUserChangeNick, 127
getNewNick, 128
getOldNick, 128
getUserId, 128
shared::events::eventUserChangeStatus, 129
eventUserChangeStatus, 132
getNewStatus, 133
getOldStatus, 133
getUserId, 133
shared::events::eventUserJoin, 134
eventUserJoin, 136
getUser, 136
getUserId, 137
shared::events::eventUserLeave, 138
eventUserLeave, 140
getUserId, 140
show
client::clsNotify, 78
showExpanded
MainWindow, 215
showMessage
client::MainWindow, 225
slot_netwManagerFinished
client::ircUser, 181
sltConnected
server::ircClient, 162
sock
client::clsServerConn, 95
server::ircClient, 165
server::uiClient, 276
srvConnected
client::clsServerConn, 91
srvDisconnected
client::clsServerConn, 92
stackDeep
org::kde::necessitas::origo::QtApplication, 268
standard
client::ircUser, 182
server::ircUser, 174
startApp
org::kde::necessitas::origo::QtActivity, 253
startListen
server::uiServer, 286
startupComplete
server::ircClient, 162
status
client::ircUser, 183
server::ircUser, 174
server::nickAndStatus, 231
super_dispatchKeyEvent
org::kde::necessitas::origo::QtActivity, 253
super_dispatchPopulateAccessibilityEvent
org::kde::necessitas::origo::QtActivity, 253

super_dispatchTouchEvent
org::kde::necessitas::origo::QtActivity, 253
super_dispatchTrackballEvent
org::kde::necessitas::origo::QtActivity, 254
super_onActivityResult
org::kde::necessitas::origo::QtActivity, 254
super_onApplyThemeResource
org::kde::necessitas::origo::QtActivity, 254
super_onAttachedToWindow
org::kde::necessitas::origo::QtActivity, 254
super_onBackPressed
org::kde::necessitas::origo::QtActivity, 254
super_onChildTitleChanged
org::kde::necessitas::origo::QtActivity, 254
super_onConfigurationChanged
org::kde::necessitas::origo::QtActivity, 254
super_onContentChanged
org::kde::necessitas::origo::QtActivity, 254
super_onContextItemSelected
org::kde::necessitas::origo::QtActivity, 254
super_onContextMenuClosed
org::kde::necessitas::origo::QtActivity, 254
super_onCreateContextMenu
org::kde::necessitas::origo::QtActivity, 254
super_onCreateDescription
org::kde::necessitas::origo::QtActivity, 255
super_onCreateDialog
org::kde::necessitas::origo::QtActivity, 255
super_onCreateOptionsMenu
org::kde::necessitas::origo::QtActivity, 255
super_onCreatePanelMenu
org::kde::necessitas::origo::QtActivity, 255
super_onCreatePanelView
org::kde::necessitas::origo::QtActivity, 255
super_onCreateThumbnail
org::kde::necessitas::origo::QtActivity, 255
super_onCreateView
org::kde::necessitas::origo::QtActivity, 255
super_onDetachedFromWindow
org::kde::necessitas::origo::QtActivity, 255
super_onKeyDown
org::kde::necessitas::origo::QtActivity, 255
super_onKeyLongPress
org::kde::necessitas::origo::QtActivity, 255
super_onKeyMultiple
org::kde::necessitas::origo::QtActivity, 255
super_onKeyUp
org::kde::necessitas::origo::QtActivity, 256
super_onMenuItemSelected
org::kde::necessitas::origo::QtActivity, 256
super_onMenuOpened
org::kde::necessitas::origo::QtActivity, 256
super_onNewIntent
org::kde::necessitas::origo::QtActivity, 256
super_onOptionsItemSelected
org::kde::necessitas::origo::QtActivity, 256
super_onOptionsMenuClosed
org::kde::necessitas::origo::QtActivity, 256
super_onPanelClosed
org::kde::necessitas::origo::QtActivity, 256
super_onPrepareDialog
org::kde::necessitas::origo::QtActivity, 256
super_onPrepareOptionsMenu
org::kde::necessitas::origo::QtActivity, 256
super_onPreparePanel
org::kde::necessitas::origo::QtActivity, 256
super_onRestoreInstanceState
org::kde::necessitas::origo::QtActivity, 256
super_onRetainNonConfigurationInstance
org::kde::necessitas::origo::QtActivity, 257
super_onSaveInstanceState
org::kde::necessitas::origo::QtActivity, 257
super_onSearchRequested
org::kde::necessitas::origo::QtActivity, 257
super_onTitleChanged
org::kde::necessitas::origo::QtActivity, 257
super_onTouchEvent
org::kde::necessitas::origo::QtActivity, 257
super_onTrackballEvent
org::kde::necessitas::origo::QtActivity, 257
super_onUserInteraction
org::kde::necessitas::origo::QtActivity, 257
super_onUserLeaveHint
org::kde::necessitas::origo::QtActivity, 257
super_onWindowAttributesChanged
org::kde::necessitas::origo::QtActivity, 257
super_onWindowFocusChanged
org::kde::necessitas::origo::QtActivity, 257
suppressDisconnectSignal
server::ircClient, 165

TAG
server::mainApp, 212
tag
K4U::clsLog, 76
timeBetweenPings
server::ircClient, 165
timeOfEvent
shared::events::clsEvent, 69
timeOfMessage
client::chatEntry, 52
client::chatNotification, 62
timeOutChecker
server::ircClient, 165
toJsonString
K4U::jsonCommand, 195
toVariant
shared::events::clsEvent, 68
toVariantMap
server::ircUser, 173
type
client::chatBox::content, 115

uColor
client::ircUser, 183
server::ircUser, 174
Ui, 37

ui
client::chatBox, 45
client::chatEntry, 52
client::chatNotification, 62
client::clsUserEntry, 110
client::MainWindow, 230
MainWindow, 215
server::mainApp, 212
uiClient
server::uiClient, 271
uiSendChat
server::mainApp, 207
uiServer
server::uiServer, 279
user
client::clsUserEntry, 111
userChangeNick
client::clsServerConn, 92
client::MainWindow, 225
server::ircClient, 162
server::mainApp, 207
userChangeStatus
server::ircClient, 163
server::mainApp, 208
userEnter
client::clsServerConn, 93
client::MainWindow, 226
userLeave
client::clsServerConn, 93
client::MainWindow, 227
userOffline
server::ircClient, 163
server::mainApp, 209
userOnline
server::ircClient, 164
server::mainApp, 210
userQuery
server::uiClient, 276
userQueryCompleted
client::clsServerConn, 93
client::MainWindow, 228
userStatusChange
client::clsServerConn, 94
client::MainWindow, 228
users
client::clsUserList, 115
client::MainWindow, 230
server::clsSettings, 106
server::ircUserList, 189
server::mainApp, 212

verticalLayout
client::clsUserList, 115
verticalSpacer
client::clsUserList, 115

WIDGETTYPES_CHATENTRY
client::chatBox, 42
WIDGETTYPES_NOTIFICATION
client::chatBox, 42
widget
client::chatBox::content, 115
widgetTypes
client::chatBox, 42
widthOfNick
client::chatBox, 45