

Battleship

Generated by Doxygen 1.9.1

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 AudioPlayer Class Reference	7
4.1.1 Member Enumeration Documentation	7
4.1.1.1 Clip	7
4.1.2 Member Function Documentation	8
4.1.2.1 play() [1/2]	8
4.1.2.2 play() [2/2]	8
4.2 Battleship Class Reference	9
4.2.1 Member Function Documentation	9
4.2.1.1 OnInit()	9
4.3 BattleshipException Class Reference	10
4.3.1 Constructor & Destructor Documentation	10
4.3.1.1 BattleshipException()	10
4.3.2 Member Function Documentation	10
4.3.2.1 what()	11
4.4 CallShot Class Reference	11
4.4.1 Constructor & Destructor Documentation	12
4.4.1.1 CallShot()	12
4.4.2 Member Function Documentation	12
4.4.2.1 getPosition()	12
4.5 ClientNetworkManager Class Reference	12
4.5.1 Member Function Documentation	12
4.5.1.1 init()	12
4.5.1.2 parseResponse()	13
4.5.1.3 sendRequest()	13
4.6 ClientRequest Class Reference	14
4.6.1 Constructor & Destructor Documentation	15
4.6.1.1 ~ClientRequest()	15
4.6.1.2 ClientRequest()	15
4.6.2 Member Function Documentation	15
4.6.2.1 getPlayerId()	15
4.6.2.2 getRequestType()	16
4.7 ConnectionPanel Class Reference	16
4.7.1 Constructor & Destructor Documentation	17

4.7.1.1 ConnectionPanel()	17
4.7.2 Member Function Documentation	17
4.7.2.1 getServerAddress()	17
4.7.2.2 getServerPort()	18
4.7.2.3 getUsername()	18
4.7.2.4 onConnectButtonClicked()	19
4.8 Coordinate Struct Reference	20
4.8.1 Member Function Documentation	20
4.8.1.1 operator<=>()	20
4.8.2 Member Data Documentation	20
4.8.2.1 x	21
4.8.2.2 y	21
4.9 EmoteEvent Class Reference	21
4.9.1 Constructor & Destructor Documentation	21
4.9.1.1 EmoteEvent()	22
4.9.2 Member Data Documentation	22
4.9.2.1 emote	22
4.9.2.2 playerId	22
4.10 EmoteHandler Class Reference	22
4.10.1 Detailed Description	22
4.10.2 Member Function Documentation	22
4.10.2.1 getImage()	23
4.10.2.2 getImageLarge()	23
4.10.2.3 getSound()	23
4.11 EmotePanel Class Reference	24
4.11.1 Detailed Description	24
4.11.2 Constructor & Destructor Documentation	24
4.11.2.1 EmotePanel()	24
4.12 EmotePopup Class Reference	25
4.12.1 Detailed Description	25
4.12.2 Constructor & Destructor Documentation	25
4.12.2.1 EmotePopup()	25
4.13 ErrorResponse Class Reference	26
4.13.1 Constructor & Destructor Documentation	26
4.13.1.1 ErrorResponse()	26
4.13.2 Member Data Documentation	26
4.13.2.1 exception	27
4.14 GameController Class Reference	27
4.14.1 Member Function Documentation	27
4.14.1.1 callShot()	28
4.14.1.2 connectToServer()	28
4.14.1.3 enterSetupPhase()	29

4.14.1.4 gameOver()	30
4.14.1.5 getMainThreadEventHandler()	31
4.14.1.6 getSetupPanel()	31
4.14.1.7 handleGameEvent()	31
4.14.1.8 handleQuitGameEvent()	32
4.14.1.9 init()	33
4.14.1.10 playerReady()	34
4.14.1.11 quitGame()	34
4.14.1.12 sendEmote()	35
4.14.1.13 showEmote()	36
4.14.1.14 showError()	37
4.14.1.15 startGame()	37
4.15 GameEvent Class Reference	38
4.15.1 Detailed Description	39
4.15.2 Constructor & Destructor Documentation	39
4.15.2.1 GameEvent()	39
4.15.3 Member Data Documentation	39
4.15.3.1 hit	39
4.15.3.2 hitShip	39
4.15.3.3 nextPlayerId	39
4.15.3.4 playerId	40
4.15.3.5 position	40
4.15.3.6 sunk	40
4.16 GameInstance Class Reference	40
4.16.1 Member Function Documentation	40
4.16.1.1 executeShot()	40
4.16.1.2 getGameState()	41
4.16.1.3 isReady()	42
4.16.1.4 joinGame()	42
4.16.1.5 reset()	43
4.16.1.6 startGame()	44
4.17 GameOverEvent Class Reference	45
4.17.1 Constructor & Destructor Documentation	46
4.17.1.1 GameOverEvent()	46
4.17.2 Member Data Documentation	46
4.17.2.1 winnerPlayerId	46
4.18 GameState Class Reference	46
4.18.1 Member Enumeration Documentation	47
4.18.1.1 State	47
4.18.1.2 Type	47
4.18.2 Constructor & Destructor Documentation	48
4.18.2.1 GameState()	48

4.18.3 Member Function Documentation	48
4.18.3.1 addPlayer()	48
4.18.3.2 addShips()	49
4.18.3.3 finish()	49
4.18.3.4 gameOver()	50
4.18.3.5 getCurrentPlayerId()	51
4.18.3.6 getOppShipSunk()	51
4.18.3.7 getOtherPlayer()	51
4.18.3.8 getPlayer()	52
4.18.3.9 getPlayerGrid()	52
4.18.3.10 getPlayers()	53
4.18.3.11 getShip()	53
4.18.3.12 getState()	53
4.18.3.13 getWinner()	54
4.18.3.14 registerShot()	54
4.18.3.15 removePlayer()	55
4.18.3.16 shotIsLegal()	56
4.18.3.17 start()	56
4.18.3.18 updateBoards()	57
4.18.3.19 updateOppShipSunk()	58
4.19 GameWindow Class Reference	59
4.19.1 Constructor & Destructor Documentation	59
4.19.1.1 GameWindow()	60
4.19.2 Member Function Documentation	60
4.19.2.1 onClose()	60
4.19.2.2 setStatus()	60
4.19.2.3 showPanel()	61
4.20 std::hash< uuid > Struct Reference	61
4.20.1 Member Function Documentation	61
4.20.1.1 operator()()	61
4.21 ImagePanel Class Reference	62
4.21.1 Detailed Description	62
4.21.2 Constructor & Destructor Documentation	62
4.21.2.1 ImagePanel()	62
4.21.3 Member Function Documentation	63
4.21.3.1 onSize()	63
4.21.3.2 paintEvent()	63
4.22 InputField Class Reference	64
4.22.1 Constructor & Destructor Documentation	64
4.22.1.1 InputField()	64
4.22.2 Member Function Documentation	64
4.22.2.1 getValue()	65

4.23 JoinGame Class Reference	65
4.23.1 Constructor & Destructor Documentation	65
4.23.1.1 JoinGame()	66
4.23.2 Member Function Documentation	66
4.23.2.1 getPlayerName()	66
4.24 JoinGameSuccess Class Reference	66
4.24.1 Detailed Description	67
4.24.2 Constructor & Destructor Documentation	67
4.24.2.1 JoinGameSuccess()	67
4.24.3 Member Function Documentation	67
4.24.3.1 wasSuccessful()	67
4.25 Logger Class Reference	67
4.25.1 Detailed Description	68
4.25.2 Member Function Documentation	68
4.25.2.1 log() [1/2]	68
4.25.2.2 log() [2/2]	69
4.25.2.3 setPrefix()	69
4.26 MainGamePanel Class Reference	70
4.26.1 Constructor & Destructor Documentation	70
4.26.1.1 MainGamePanel()	70
4.26.2 Member Function Documentation	70
4.26.2.1 buildGameState()	70
4.26.2.2 displayEmote()	71
4.27 PlacementGrid Class Reference	72
4.27.1 Constructor & Destructor Documentation	73
4.27.1.1 PlacementGrid()	73
4.27.2 Member Function Documentation	73
4.27.2.1 displayGrid()	73
4.27.2.2 highlightTiles()	74
4.27.2.3 OnMouseClicked()	75
4.27.2.4 OnMouseMotion()	76
4.27.3 Member Data Documentation	76
4.27.3.1 cellX_prev	76
4.27.3.2 cellY_prev	76
4.28 PlayAgain Class Reference	77
4.28.1 Constructor & Destructor Documentation	77
4.28.1.1 PlayAgain()	77
4.29 Player Class Reference	77
4.29.1 Constructor & Destructor Documentation	78
4.29.1.1 Player()	78
4.29.2 Member Function Documentation	78
4.29.2.1 getId()	78

4.29.2.2 getName()	79
4.29.2.3 operator==()	79
4.30 PlayerGrid Class Reference	79
4.30.1 Detailed Description	80
4.30.2 Constructor & Destructor Documentation	80
4.30.2.1 PlayerGrid()	80
4.30.3 Member Data Documentation	80
4.30.3.1 playerId	80
4.30.3.2 shipsPlaced	80
4.30.3.3 shotsFired	80
4.30.3.4 shotsReceived	81
4.31 QuitGame Class Reference	81
4.31.1 Constructor & Destructor Documentation	81
4.31.1.1 QuitGame()	81
4.32 QuitGameEvent Class Reference	82
4.32.1 Constructor & Destructor Documentation	82
4.32.1.1 QuitGameEvent()	82
4.32.2 Member Data Documentation	82
4.32.2.1 quitPlayerId	83
4.33 RequestHandler Class Reference	83
4.33.1 Member Function Documentation	83
4.33.1.1 handleRequest()	83
4.34 ResponseListenerThread Class Reference	84
4.34.1 Detailed Description	85
4.34.2 Constructor & Destructor Documentation	85
4.34.2.1 ResponseListenerThread()	85
4.34.3 Member Function Documentation	85
4.34.3.1 Entry()	85
4.35 SendEmote Class Reference	86
4.35.1 Constructor & Destructor Documentation	86
4.35.1.1 SendEmote()	86
4.35.2 Member Function Documentation	86
4.35.2.1 getEmote()	87
4.36 ServerNetworkManager Class Reference	87
4.36.1 Constructor & Destructor Documentation	87
4.36.1.1 ServerNetworkManager()	87
4.36.1.2 ~ServerNetworkManager()	88
4.36.2 Member Function Documentation	88
4.36.2.1 broadcastMessage()	88
4.36.2.2 listenerLoop()	88
4.36.2.3 onPlayerLeft()	89
4.37 ServerResponse Class Reference	89

4.37.1 Constructor & Destructor Documentation	90
4.37.1.1 ~ServerResponse()	90
4.37.1.2 ServerResponse()	90
4.37.2 Member Function Documentation	90
4.37.2.1 operator<=>()	90
4.37.3 Member Data Documentation	90
4.37.3.1 responseType	91
4.38 SetupManager Class Reference	91
4.38.1 Constructor & Destructor Documentation	91
4.38.1.1 SetupManager()	91
4.38.2 Member Function Documentation	92
4.38.2.1 getGrid()	92
4.38.2.2 placedAllShips()	92
4.38.2.3 placeShip()	92
4.38.3 Member Data Documentation	93
4.38.3.1 _selectedShip	93
4.38.3.2 _ships_placed	94
4.39 SetupPanel Class Reference	94
4.39.1 Constructor & Destructor Documentation	94
4.39.1.1 SetupPanel()	94
4.39.2 Member Function Documentation	95
4.39.2.1 getReadyButton()	95
4.39.2.2 getReadyText()	96
4.39.2.3 getShipButton()	96
4.39.2.4 OnKeyDown()	96
4.39.2.5 OnReadyButtonClicked()	97
4.40 Ship Class Reference	98
4.40.1 Member Enumeration Documentation	99
4.40.1.1 Orientation	99
4.40.2 Constructor & Destructor Documentation	99
4.40.2.1 Ship()	99
4.40.3 Member Function Documentation	99
4.40.3.1 getId()	99
4.40.3.2 getLength()	100
4.40.3.3 getOrientation()	100
4.40.3.4 getPosition()	100
4.40.3.5 hasSunken()	100
4.40.3.6 hit()	101
4.40.3.7 setOrientation()	101
4.40.3.8 setPosition()	102
4.41 ShipPanel Class Reference	102
4.41.1 Constructor & Destructor Documentation	102

4.41.1.1 ShipPanel()	102
4.41.2 Member Function Documentation	103
4.41.2.1 update()	103
4.42 StartGame Class Reference	103
4.42.1 Constructor & Destructor Documentation	104
4.42.1.1 StartGame()	104
4.42.2 Member Function Documentation	104
4.42.2.1 getShips()	104
4.43 StartGameSuccess Class Reference	105
4.43.1 Constructor & Destructor Documentation	105
4.43.1.1 StartGameSuccess()	105
4.43.2 Member Data Documentation	105
4.43.2.1 players	106
4.43.2.2 startingPlayerId	106
4.44 uuid Class Reference	106
4.44.1 Constructor & Destructor Documentation	106
4.44.1.1 uuid() [1/2]	106
4.44.1.2 uuid() [2/2]	106
4.44.2 Member Function Documentation	107
4.44.2.1 generateRandomUuid()	107
4.44.2.2 operator==()	107
4.44.2.3 ToString()	107
4.45 ViewGrid Class Reference	108
4.45.1 Member Enumeration Documentation	108
4.45.1.1 GridType	108
4.45.2 Constructor & Destructor Documentation	109
4.45.2.1 ViewGrid()	109
4.45.3 Member Function Documentation	109
4.45.3.1 showShips()	109
4.45.3.2 showShots()	110
5 File Documentation	111
5.1 /home/nico/Desktop/battleship/src/client/AudioPlayer.cpp File Reference	111
5.2 /home/nico/Desktop/battleship/src/client/AudioPlayer.h File Reference	111
5.3 /home/nico/Desktop/battleship/src/client/Battleship.cpp File Reference	111
5.4 /home/nico/Desktop/battleship/src/client/Battleship.h File Reference	112
5.5 /home/nico/Desktop/battleship/src/client/ClientNetworkManager.cpp File Reference	112
5.6 /home/nico/Desktop/battleship/src/client/ClientNetworkManager.h File Reference	112
5.7 /home/nico/Desktop/battleship/src/client/EmoteHandler.cpp File Reference	112
5.8 /home/nico/Desktop/battleship/src/client/EmoteHandler.h File Reference	113
5.9 /home/nico/Desktop/battleship/src/client/GameController.cpp File Reference	113
5.10 /home/nico/Desktop/battleship/src/client/GameController.h File Reference	113

5.11 /home/nico/Desktop/battleship/src/client/GameWindow.cpp File Reference	114
5.12 /home/nico/Desktop/battleship/src/client/GameWindow.h File Reference	114
5.13 /home/nico/Desktop/battleship/src/client/main.cpp File Reference	114
5.13.1 Function Documentation	114
5.13.1.1 wxIMPLEMENT_APP()	114
5.14 /home/nico/Desktop/battleship/src/server/main.cpp File Reference	114
5.14.1 Function Documentation	115
5.14.1.1 main()	115
5.15 /home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.cpp File Reference	115
5.16 /home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.h File Reference	115
5.17 /home/nico/Desktop/battleship/src/client/panels/MainGamePanel.cpp File Reference	116
5.18 /home/nico/Desktop/battleship/src/client/panels/MainGamePanel.h File Reference	116
5.19 /home/nico/Desktop/battleship/src/client/panels/SetupPanel.cpp File Reference	116
5.20 /home/nico/Desktop/battleship/src/client/panels/SetupPanel.h File Reference	116
5.21 /home/nico/Desktop/battleship/src/client/ResponseListenerThread.cpp File Reference	117
5.22 /home/nico/Desktop/battleship/src/client/ResponseListenerThread.h File Reference	117
5.23 /home/nico/Desktop/battleship/src/client/SetupManager.cpp File Reference	117
5.24 /home/nico/Desktop/battleship/src/client/SetupManager.h File Reference	117
5.25 /home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.cpp File Reference	118
5.26 /home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.h File Reference	118
5.27 /home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.cpp File Reference	118
5.28 /home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.h File Reference	118
5.29 /home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.cpp File Reference	118
5.30 /home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.h File Reference	119
5.31 /home/nico/Desktop/battleship/src/client/uiElements/InputField.cpp File Reference	119
5.32 /home/nico/Desktop/battleship/src/client/uiElements/InputField.h File Reference	119
5.33 /home/nico/Desktop/battleship/src/client/uiElements/PlacementGrid.cpp File Reference	119
5.34 /home/nico/Desktop/battleship/src/client/uiElements/PlacementGrid.h File Reference	119
5.35 /home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.cpp File Reference	120
5.36 /home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.h File Reference	120
5.37 /home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.cpp File Reference	120
5.38 /home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.h File Reference	120
5.39 /home/nico/Desktop/battleship/src/common/exceptions/BattleshipException.h File Reference	121
5.40 /home/nico/Desktop/battleship/src/common/game_state/Coordinate.h File Reference	121
5.41 /home/nico/Desktop/battleship/src/common/game_state/GameState.cpp File Reference	121
5.42 /home/nico/Desktop/battleship/src/common/game_state/GameState.h File Reference	121
5.43 /home/nico/Desktop/battleship/src/common/game_state/Player.cpp File Reference	122
5.44 /home/nico/Desktop/battleship/src/common/game_state/Player.h File Reference	122
5.45 /home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.cpp File Reference	122
5.46 /home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.h File Reference	122
5.47 /home/nico/Desktop/battleship/src/common/game_state/Ship.cpp File Reference	122
5.48 /home/nico/Desktop/battleship/src/common/game_state/Ship.h File Reference	123

5.49	/home/nico/Desktop/battleship/src/common/Logger.cpp File Reference	123
5.50	/home/nico/Desktop/battleship/src/common/Logger.h File Reference	123
5.50.1	Macro Definition Documentation	123
5.50.1.1	LOG	124
5.51	/home/nico/Desktop/battleship/src/common/network/requests/CallShot.cpp File Reference	124
5.52	/home/nico/Desktop/battleship/src/common/network/requests/CallShot.h File Reference	124
5.53	/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.cpp File Reference	124
5.54	/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.h File Reference	124
5.54.1	Enumeration Type Documentation	125
5.54.1.1	RequestType	125
5.55	/home/nico/Desktop/battleship/src/common/network/requests/JoinGame.cpp File Reference	125
5.56	/home/nico/Desktop/battleship/src/common/network/requests/JoinGame.h File Reference	125
5.57	/home/nico/Desktop/battleship/src/common/network/requests/PlayAgain.cpp File Reference	126
5.58	/home/nico/Desktop/battleship/src/common/network/requests/PlayAgain.h File Reference	126
5.59	/home/nico/Desktop/battleship/src/common/network/requests/QuitGame.cpp File Reference	126
5.60	/home/nico/Desktop/battleship/src/common/network/requests/QuitGame.h File Reference	126
5.61	/home/nico/Desktop/battleship/src/common/network/requests/SendEmote.cpp File Reference	126
5.62	/home/nico/Desktop/battleship/src/common/network/requests/SendEmote.h File Reference	127
5.63	/home/nico/Desktop/battleship/src/common/network/requests/StartGame.cpp File Reference	127
5.64	/home/nico/Desktop/battleship/src/common/network/requests/StartGame.h File Reference	127
5.65	/home/nico/Desktop/battleship/src/common/network/responses/EmotionEvent.cpp File Reference	127
5.66	/home/nico/Desktop/battleship/src/common/network/responses/EmotionEvent.h File Reference	128
5.66.1	Enumeration Type Documentation	128
5.66.1.1	EmoteType	128
5.67	/home/nico/Desktop/battleship/src/common/network/responses/ErrorResponse.cpp File Reference	128
5.68	/home/nico/Desktop/battleship/src/common/network/responses/ErrorResponse.h File Reference	129
5.69	/home/nico/Desktop/battleship/src/common/network/responses/GameEvent.cpp File Reference	129
5.70	/home/nico/Desktop/battleship/src/common/network/responses/GameEvent.h File Reference	129
5.71	/home/nico/Desktop/battleship/src/common/network/responses/GameOverEvent.cpp File Reference	129
5.72	/home/nico/Desktop/battleship/src/common/network/responses/GameOverEvent.h File Reference	130
5.73	/home/nico/Desktop/battleship/src/common/network/responses/JoinGameSuccess.cpp File Reference	130
5.74	/home/nico/Desktop/battleship/src/common/network/responses/JoinGameSuccess.h File Reference	130
5.75	/home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.cpp File Reference	130
5.76	/home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.h File Reference	130
5.77	/home/nico/Desktop/battleship/src/common/network/responses/ServerResponse.cpp File Reference	131
5.78	/home/nico/Desktop/battleship/src/common/network/responses/ServerResponse.h File Reference	131
5.78.1	Enumeration Type Documentation	131
5.78.1.1	ResponseType	131
5.79	/home/nico/Desktop/battleship/src/common/network/responses/StartGameSuccess.cpp File Reference	132
5.80	/home/nico/Desktop/battleship/src/common/network/responses/StartGameSuccess.h File Reference	132
5.81	/home/nico/Desktop/battleship/src/common/serialization/serialization.h File Reference	132

5.81.1 Function Documentation	133
5.81.1.1 from_json() [1/2]	133
5.81.1.2 from_json() [2/2]	133
5.81.1.3 NLOHMANN_JSON_SERIALIZE_ENUM()	133
5.81.1.4 to_json() [1/2]	133
5.81.1.5 to_json() [2/2]	133
5.82 /home/nico/Desktop/battleship/src/common/uuid.cpp File Reference	134
5.83 /home/nico/Desktop/battleship/src/common/uuid.h File Reference	134
5.84 /home/nico/Desktop/battleship/src/server/GameInstance.cpp File Reference	134
5.85 /home/nico/Desktop/battleship/src/server/GameInstance.h File Reference	134
5.86 /home/nico/Desktop/battleship/src/server/RequestHandler.cpp File Reference	135
5.87 /home/nico/Desktop/battleship/src/server/RequestHandler.h File Reference	135
5.88 /home/nico/Desktop/battleship/src/server/ServerNetworkManager.cpp File Reference	136
5.89 /home/nico/Desktop/battleship/src/server/ServerNetworkManager.h File Reference	136
Index	137

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

AudioPlayer	7
ClientNetworkManager	12
ClientRequest	14
CallShot	11
JoinGame	65
PlayAgain	77
QuitGame	81
SendEmote	86
StartGame	103
Coordinate	20
EmoteHandler	22
std::exception	
BattleshipException	10
GameController	27
GameInstance	40
GameState	46
std::hash< uuid >	61
Logger	67
Player	77
PlayerGrid	79
RequestHandler	83
ServerNetworkManager	87
ServerResponse	89
EmoteEvent	21
ErrorResponse	26
GameEvent	38
GameOverEvent	45
JoinGameSuccess	66
QuitGameEvent	82
StartGameSuccess	105
SetupManager	91
Ship	98
uuid	106
wxApp	
Battleship	9

wxFrame	
GameWindow	59
wxPanel	
ConnectionPanel	16
EmotePanel	24
ImagePanel	62
InputField	64
MainGamePanel	70
PlacementGrid	72
SetupPanel	94
ShipPanel	102
ViewGrid	108
wxPopupWindow	
EmotePopup	25
wxThread	
ResponseListenerThread	84

Chapter 2

Class Index

2.1 Class List

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

AudioPlayer	7
Battleship	9
BattleshipException	10
CallShot	11
ClientNetworkManager	12
ClientRequest	14
ConnectionPanel	16
Coordinate	20
EmoteEvent	21
EmoteHandler	22
EmotePanel	24
EmotePopup	25
ErrorResponse	26
GameController	27
GameEvent	38
GameInstance	40
GameOverEvent	45
GameState	46
GameWindow	59
std::hash< uuid >	61
ImagePanel	62
InputField	64
JoinGame	65
JoinGameSuccess	66
Logger	67
MainGamePanel	70
PlacementGrid	72
PlayAgain	77
Player	77
PlayerGrid	79
QuitGame	81
QuitGameEvent	82
RequestHandler	83
ResponseListenerThread	84
SendEmote	86

ServerNetworkManager	87
ServerResponse	89
SetupManager	91
SetupPanel	94
Ship	98
ShipPanel	102
StartGame	103
StartGameSuccess	105
uuid	106
ViewGrid	108

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

/home/nico/Desktop/battleship/src/client/ AudioPlayer.cpp	111
/home/nico/Desktop/battleship/src/client/ AudioPlayer.h	111
/home/nico/Desktop/battleship/src/client/ Battleship.cpp	111
/home/nico/Desktop/battleship/src/client/ Battleship.h	112
/home/nico/Desktop/battleship/src/client/ ClientNetworkManager.cpp	112
/home/nico/Desktop/battleship/src/client/ ClientNetworkManager.h	112
/home/nico/Desktop/battleship/src/client/ EmoteHandler.cpp	112
/home/nico/Desktop/battleship/src/client/ EmoteHandler.h	113
/home/nico/Desktop/battleship/src/client/ GameController.cpp	113
/home/nico/Desktop/battleship/src/client/ GameController.h	113
/home/nico/Desktop/battleship/src/client/ GameWindow.cpp	114
/home/nico/Desktop/battleship/src/client/ GameWindow.h	114
/home/nico/Desktop/battleship/src/client/ main.cpp	114
/home/nico/Desktop/battleship/src/client/ ResponseListenerThread.cpp	117
/home/nico/Desktop/battleship/src/client/ ResponseListenerThread.h	117
/home/nico/Desktop/battleship/src/client/ SetupManager.cpp	117
/home/nico/Desktop/battleship/src/client/ SetupManager.h	117
/home/nico/Desktop/battleship/src/client/panels/ ConnectionPanel.cpp	115
/home/nico/Desktop/battleship/src/client/panels/ ConnectionPanel.h	115
/home/nico/Desktop/battleship/src/client/panels/ MainGamePanel.cpp	116
/home/nico/Desktop/battleship/src/client/panels/ MainGamePanel.h	116
/home/nico/Desktop/battleship/src/client/panels/ SetupPanel.cpp	116
/home/nico/Desktop/battleship/src/client/panels/ SetupPanel.h	116
/home/nico/Desktop/battleship/src/client/uiElements/ EmotePanel.cpp	118
/home/nico/Desktop/battleship/src/client/uiElements/ EmotePanel.h	118
/home/nico/Desktop/battleship/src/client/uiElements/ EmotePopup.cpp	118
/home/nico/Desktop/battleship/src/client/uiElements/ EmotePopup.h	118
/home/nico/Desktop/battleship/src/client/uiElements/ ImagePanel.cpp	118
/home/nico/Desktop/battleship/src/client/uiElements/ ImagePanel.h	119
/home/nico/Desktop/battleship/src/client/uiElements/ InputField.cpp	119
/home/nico/Desktop/battleship/src/client/uiElements/ InputField.h	119
/home/nico/Desktop/battleship/src/client/uiElements/ PlacementGrid.cpp	119
/home/nico/Desktop/battleship/src/client/uiElements/ PlacementGrid.h	119
/home/nico/Desktop/battleship/src/client/uiElements/ ShipPanel.cpp	120
/home/nico/Desktop/battleship/src/client/uiElements/ ShipPanel.h	120

/home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.cpp	120
/home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.h	120
/home/nico/Desktop/battleship/src/common/Logger.cpp	123
/home/nico/Desktop/battleship/src/common/Logger.h	123
/home/nico/Desktop/battleship/src/common/uuid.cpp	134
/home/nico/Desktop/battleship/src/common/uuid.h	134
/home/nico/Desktop/battleship/src/common/exceptions/BattleshipException.h	121
/home/nico/Desktop/battleship/src/common/game_state/Coordinate.h	121
/home/nico/Desktop/battleship/src/common/game_state/GameState.cpp	121
/home/nico/Desktop/battleship/src/common/game_state/GameState.h	121
/home/nico/Desktop/battleship/src/common/game_state/Player.cpp	122
/home/nico/Desktop/battleship/src/common/game_state/Player.h	122
/home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.cpp	122
/home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.h	122
/home/nico/Desktop/battleship/src/common/game_state/Ship.cpp	122
/home/nico/Desktop/battleship/src/common/game_state/Ship.h	123
/home/nico/Desktop/battleship/src/common/network/requests/CallShot.cpp	124
/home/nico/Desktop/battleship/src/common/network/requests/CallShot.h	124
/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.cpp	124
/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.h	124
/home/nico/Desktop/battleship/src/common/network/requests/JoinGame.cpp	125
/home/nico/Desktop/battleship/src/common/network/requests/JoinGame.h	125
/home/nico/Desktop/battleship/src/common/network/requests/PlayAgain.cpp	126
/home/nico/Desktop/battleship/src/common/network/requests/PlayAgain.h	126
/home/nico/Desktop/battleship/src/common/network/requests/QuitGame.cpp	126
/home/nico/Desktop/battleship/src/common/network/requests/QuitGame.h	126
/home/nico/Desktop/battleship/src/common/network/requests/SendEmote.cpp	126
/home/nico/Desktop/battleship/src/common/network/requests/SendEmote.h	127
/home/nico/Desktop/battleship/src/common/network/requests/StartGame.cpp	127
/home/nico/Desktop/battleship/src/common/network/requests/StartGame.h	127
/home/nico/Desktop/battleship/src/common/network/responses/EmoteEvent.cpp	127
/home/nico/Desktop/battleship/src/common/network/responses/EmoteEvent.h	128
/home/nico/Desktop/battleship/src/common/network/responses/ErrorResponse.cpp	128
/home/nico/Desktop/battleship/src/common/network/responses/ErrorResponse.h	129
/home/nico/Desktop/battleship/src/common/network/responses/GameEvent.cpp	129
/home/nico/Desktop/battleship/src/common/network/responses/GameEvent.h	129
/home/nico/Desktop/battleship/src/common/network/responses/GameOverEvent.cpp	129
/home/nico/Desktop/battleship/src/common/network/responses/GameOverEvent.h	130
/home/nico/Desktop/battleship/src/common/network/responses/JoinGameSuccess.cpp	130
/home/nico/Desktop/battleship/src/common/network/responses/JoinGameSuccess.h	130
/home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.cpp	130
/home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.h	130
/home/nico/Desktop/battleship/src/common/network/responses/ServerResponse.cpp	131
/home/nico/Desktop/battleship/src/common/network/responses/ServerResponse.h	131
/home/nico/Desktop/battleship/src/common/network/responses/StartGameSuccess.cpp	132
/home/nico/Desktop/battleship/src/common/network/responses/StartGameSuccess.h	132
/home/nico/Desktop/battleship/src/common/serialization/serialization.h	132
/home/nico/Desktop/battleship/src/server/GameInstance.cpp	134
/home/nico/Desktop/battleship/src/server/GameInstance.h	134
/home/nico/Desktop/battleship/src/server/main.cpp	114
/home/nico/Desktop/battleship/src/server/RequestHandler.cpp	135
/home/nico/Desktop/battleship/src/server/RequestHandler.h	135
/home/nico/Desktop/battleship/src/server/ServerNetworkManager.cpp	136
/home/nico/Desktop/battleship/src/server/ServerNetworkManager.h	136

Chapter 4

Class Documentation

4.1 AudioPlayer Class Reference

```
#include <AudioPlayer.h>
```

Public Types

- enum [Clip](#) {
 [ButtonClick](#) , [SelectShip](#) , [PlaceShip](#) , [Cannon](#) ,
 [Hit](#) , [Miss](#) , [GameOver](#) , [PopUp](#) }

Static Public Member Functions

- static void [play](#) (const [Clip](#) &clip)
- static void [play](#) (const std::string &file)

4.1.1 Member Enumeration Documentation

4.1.1.1 Clip

```
enum AudioPlayer::Clip
```

Collection of general sound effect clips

Enumerator

ButtonClick	
SelectShip	
PlaceShip	
Cannon	
Hit	
Miss	
GameOver	
PopUp	

4.1.2 Member Function Documentation

4.1.2.1 play() [1/2]

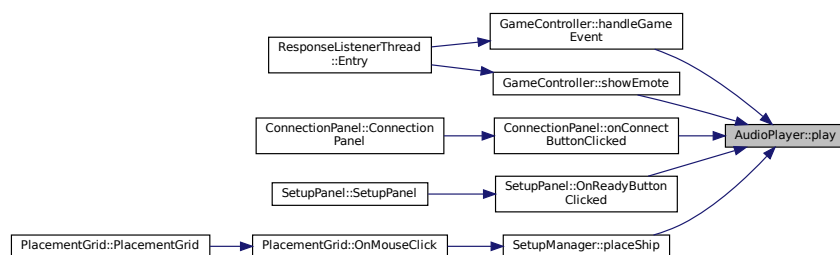
```
void AudioPlayer::play (
    const Clip & clip ) [static]
```

Play the selected audio clip

Parameters

<i>clip</i>	
-------------	--

Here is the caller graph for this function:



4.1.2.2 play() [2/2]

```
void AudioPlayer::play (
    const std::string & file ) [static]
```

Play a general audio file in .wav format. Use this to play the sound of emotes: `AudioPlayer::play(EmoteHandler::getSound(emote))`;

Parameters

<i>file</i>	full path to the .wav-file (Has to be specific WAV format! Default output of Audacity works)"
-------------	---

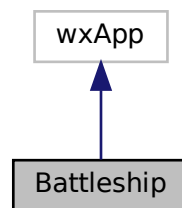
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/AudioPlayer.h](#)
- [/home/nico/Desktop/battleship/src/client/AudioPlayer.cpp](#)

4.2 Battleship Class Reference

```
#include <Battleship.h>
```

Inheritance diagram for Battleship:



Public Member Functions

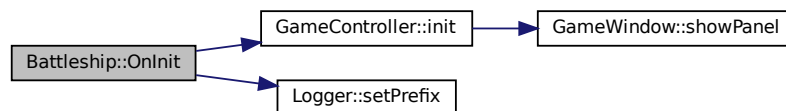
- auto `OnInit()` -> bool override

4.2.1 Member Function Documentation

4.2.1.1 OnInit()

```
auto Battleship::OnInit ( ) -> bool [override]
```

Here is the call graph for this function:



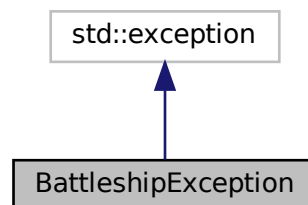
The documentation for this class was generated from the following files:

- `/home/nico/Desktop/battleship/src/client/Battleship.h`
- `/home/nico/Desktop/battleship/src/client/Battleship.cpp`

4.3 BattleshipException Class Reference

```
#include <BattleshipException.h>
```

Inheritance diagram for BattleshipException:



Public Member Functions

- [BattleshipException](#) (const std::string &message)
- const char * [what](#) () const noexcept override

4.3.1 Constructor & Destructor Documentation

4.3.1.1 BattleshipException()

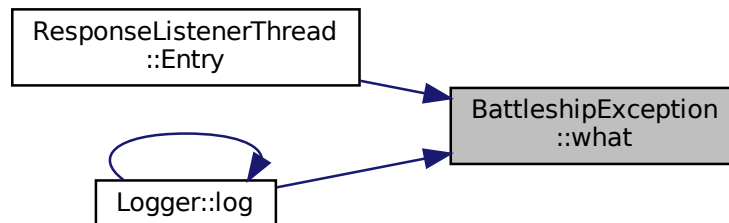
```
BattleshipException::BattleshipException (  
    const std::string & message ) [inline], [explicit]
```

4.3.2 Member Function Documentation

4.3.2.1 what()

```
const char* BattleshipException::what ( ) const [inline], [override], [noexcept]
```

Here is the caller graph for this function:



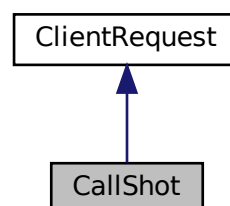
The documentation for this class was generated from the following file:

- </home/nico/Desktop/battleship/src/common/exceptions/BattleshipException.h>

4.4 CallShot Class Reference

```
#include <CallShot.h>
```

Inheritance diagram for CallShot:



Public Member Functions

- `CallShot` (`uuid` playerId, `Coordinate` position)
- auto `getPosition` () const -> `Coordinate`

Additional Inherited Members

4.4.1 Constructor & Destructor Documentation

4.4.1.1 CallShot()

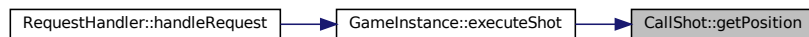
```
CallShot::CallShot (
    uuid playerId,
    Coordinate position )
```

4.4.2 Member Function Documentation

4.4.2.1 getPosition()

```
auto CallShot::getPosition ( ) const -> Coordinate
```

Here is the caller graph for this function:



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

- /home/nico/Desktop/battleship/src/common/network/requests/CallShot.h
- /home/nico/Desktop/battleship/src/common/network/requests/CallShot.cpp

4.5 ClientNetworkManager Class Reference

```
#include <ClientNetworkManager.h>
```

Static Public Member Functions

- static void [init](#) (const std::string &host, const uint16_t port)
- static void [sendRequest](#) (const [ClientRequest](#) &request)
- static std::unique_ptr< [ServerResponse](#) > [parseResponse](#) (const std::string &message)

4.5.1 Member Function Documentation

4.5.1.1 init()

```
static void ClientNetworkManager::init (
    const std::string & host,
    const uint16_t port ) [static]
```

establishes a connection to the server specified by host address and port number

Parameters

<i>host</i>	server ip
<i>port</i>	server port

Here is the caller graph for this function:



4.5.1.2 parseResponse()

```
std::unique_ptr< ServerResponse > ClientNetworkManager::parseResponse (
    const std::string & message ) [static]
```

deserializes a server response

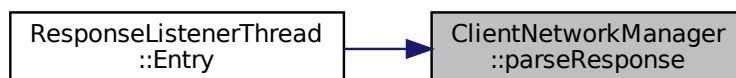
Parameters

<i>message</i>	server response in json format
----------------	--------------------------------

Returns

response object

Here is the caller graph for this function:



4.5.1.3 sendRequest()

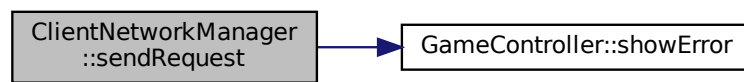
```
void ClientNetworkManager::sendRequest (
    const ClientRequest & request ) [static]
```

serializes and sends a client request to the server

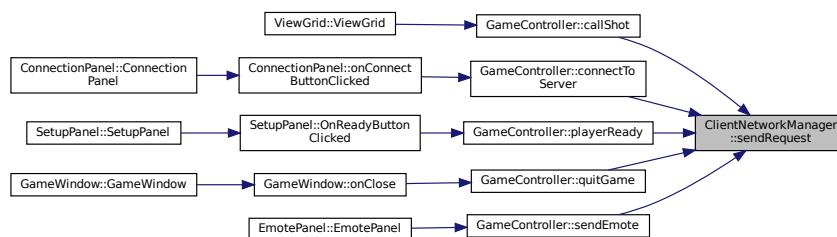
Parameters

<i>request</i>	message to the server
----------------	-----------------------

Here is the call graph for this function:



Here is the caller graph for this function:



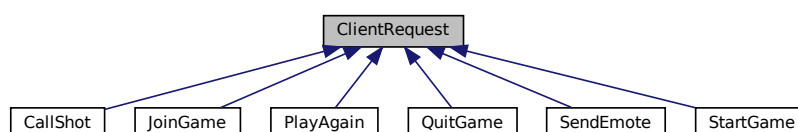
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/ClientNetworkManager.h](#)
- [/home/nico/Desktop/battleship/src/client/ClientNetworkManager.cpp](#)

4.6 ClientRequest Class Reference

```
#include <ClientRequest.h>
```

Inheritance diagram for ClientRequest:



Public Member Functions

- auto [getRequestType](#) () const -> [RequestType](#)
- auto [getPlayerId](#) () const -> [uuid](#)
- virtual [~ClientRequest](#) ()=default

Protected Member Functions

- [ClientRequest](#) ([uuid](#) playerId, [RequestType](#) requestType)

4.6.1 Constructor & Destructor Documentation

4.6.1.1 ~ClientRequest()

```
virtual ClientRequest::~~ClientRequest ( ) [virtual], [default]
```

4.6.1.2 ClientRequest()

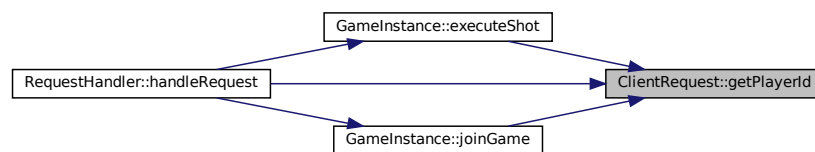
```
ClientRequest::ClientRequest (
    uuid playerId,
    RequestType requestType ) [protected]
```

4.6.2 Member Function Documentation

4.6.2.1 getPlayerId()

```
auto ClientRequest::getPlayerId ( ) const -> uuid
```

Here is the caller graph for this function:



4.6.2.2 getRequestType()

```
auto ClientRequest::getRequestType ( ) const -> RequestType
```

Here is the caller graph for this function:



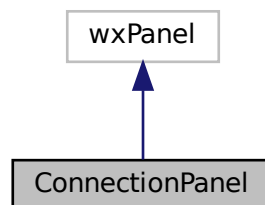
The documentation for this class was generated from the following files:

- `/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.h`
- `/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.cpp`

4.7 ConnectionPanel Class Reference

```
#include <ConnectionPanel.h>
```

Inheritance diagram for ConnectionPanel:



Public Member Functions

- `ConnectionPanel (wxWindow *parent)`
Constructor for `ConnectionPanel`.
- `void onConnectButtonClicked (wxCommandEvent &event)`
Button event handler. Will trigger `GameController::connectToServer()` to establish a connection to the server.
- `wxString getServerAddress ()`
Getter for the server address.
- `wxString getServerPort ()`
Getter for the server port.
- `wxString getUsername ()`
Getter for the username.

4.7.1 Constructor & Destructor Documentation

4.7.1.1 ConnectionPanel()

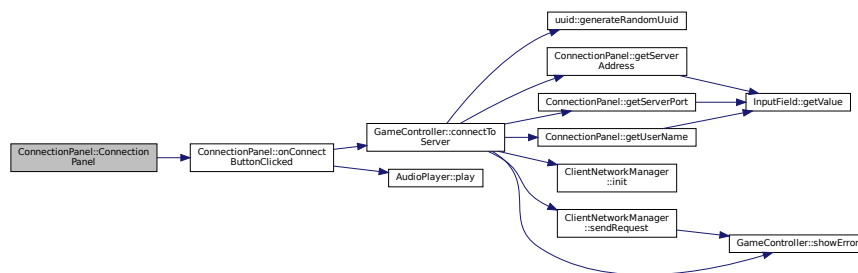
```
ConnectionPanel::ConnectionPanel (
    wxWindow * parent )
```

Constructor for [ConnectionPanel](#).

Parameters

<i>parent</i>	
---------------	--

Here is the call graph for this function:



4.7.2 Member Function Documentation

4.7.2.1 getServerAddress()

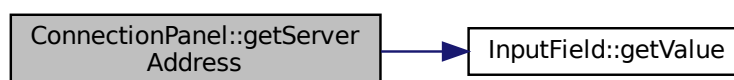
```
wxString ConnectionPanel::getServerAddress ( )
```

Getter for the server address.

Returns

wxString

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.2.2 getServerPort()

```
wxString ConnectionPanel::getServerPort ( )
```

Getter for the server port.

Returns

wxString

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.2.3 getUsername()

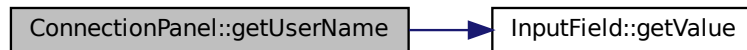
```
wxString ConnectionPanel::getUsername ( )
```

Getter for the username.

Returns

wxString

Here is the call graph for this function:



Here is the caller graph for this function:

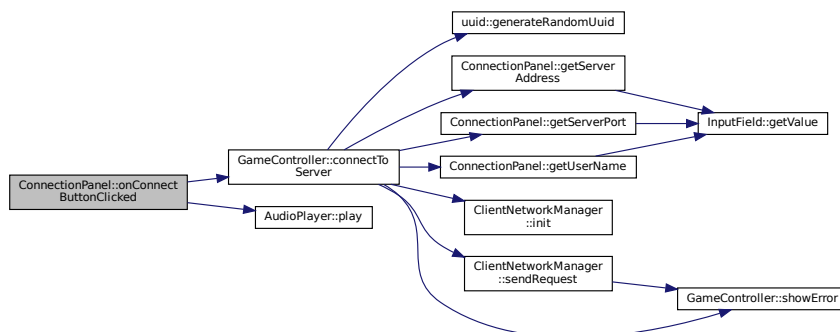


4.7.2.4 onConnectButtonClicked()

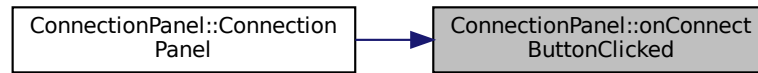
```
void ConnectionPanel::onConnectButtonClicked (
    wxCommandEvent & event )
```

Button event handler. Will trigger [GameController::connectToServer\(\)](#) to establish a connection to the server.

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:

- [/home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.h](#)
- [/home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.cpp](#)

4.8 Coordinate Struct Reference

```
#include <Coordinate.h>
```

Public Member Functions

- auto [operator<=>](#) (const [Coordinate](#) &) const =default

Public Attributes

- int [x](#) = 0
- int [y](#) = 0

4.8.1 Member Function Documentation

4.8.1.1 [operator<=>\(\)](#)

```
auto Coordinate::operator<=> (
    const Coordinate & ) const [default]
```

4.8.2 Member Data Documentation

4.8.2.1 x

```
int Coordinate::x = 0
```

4.8.2.2 y

```
int Coordinate::y = 0
```

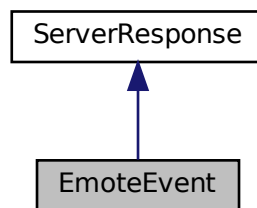
The documentation for this struct was generated from the following file:

- [/home/nico/Desktop/battleship/src/common/game_state/Coordinate.h](#)

4.9 EmoteEvent Class Reference

```
#include <EmoteEvent.h>
```

Inheritance diagram for EmoteEvent:



Public Member Functions

- [EmoteEvent](#) ([EmoteType](#) emote, [uuid](#) playerId)

Public Attributes

- const [EmoteType](#) emote
- const [uuid](#) playerId

Additional Inherited Members

4.9.1 Constructor & Destructor Documentation

4.9.1.1 EmotionEvent()

```
EmotionEvent::EmotionEvent (
    EmoteType emote,
    uuid playerId )
```

4.9.2 Member Data Documentation

4.9.2.1 emote

```
const EmoteType EmotionEvent::emote
```

4.9.2.2 playerId

```
const uuid EmotionEvent::playerId
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/[EmotionEvent.h](#)
- /home/nico/Desktop/battleship/src/common/network/responses/[EmotionEvent.cpp](#)

4.10 EmoteHandler Class Reference

```
#include <EmoteHandler.h>
```

Static Public Member Functions

- static std::string [getSound](#) (EmoteType emote)
- static std::string [getImage](#) (EmoteType emote)
- static std::string [getImageLarge](#) (EmoteType emote)

4.10.1 Detailed Description

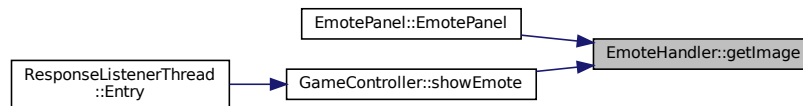
Class to provide images and audio files for all available emotes

4.10.2 Member Function Documentation

4.10.2.1 getImage()

```
std::string EmoteHandler::getImage (
    EmoteType emote ) [static]
```

Here is the caller graph for this function:



4.10.2.2 getImageLarge()

```
std::string EmoteHandler::getImageLarge (
    EmoteType emote ) [static]
```

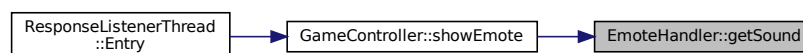
Here is the caller graph for this function:



4.10.2.3 getSound()

```
std::string EmoteHandler::getSound (
    EmoteType emote ) [static]
```

Here is the caller graph for this function:



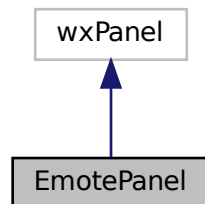
The documentation for this class was generated from the following files:

- `/home/nico/Desktop/battleship/src/client/EmoteHandler.h`
- `/home/nico/Desktop/battleship/src/client/EmoteHandler.cpp`

4.11 EmotePanel Class Reference

```
#include <EmotePanel.h>
```

Inheritance diagram for EmotePanel:



Public Member Functions

- [EmotePanel](#) (`wxWindow *parent`, `wxPoint pos`)

4.11.1 Detailed Description

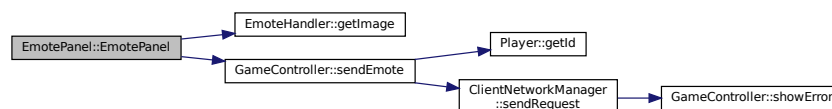
Panel to display the available emotes the user can use to communicate with the opponent

4.11.2 Constructor & Destructor Documentation

4.11.2.1 EmotePanel()

```
EmotePanel::EmotePanel (
    wxWindow * parent,
    wxPoint pos )
```

Here is the call graph for this function:



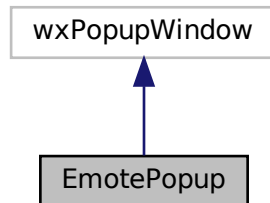
The documentation for this class was generated from the following files:

- `/home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.h`
- `/home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.cpp`

4.12 EmotePopup Class Reference

```
#include <EmotePopup.h>
```

Inheritance diagram for EmotePopup:



Public Member Functions

- [EmotePopup](#) (`wxWindow *parent`, `wxPoint pos`, [EmoteType](#) `emote`)

4.12.1 Detailed Description

Popup window to display emotes on screen. They disappear when clicking on the image.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 EmotePopup()

```
EmotePopup::EmotePopup (
    wxWindow * parent,
    wxPoint pos,
    EmoteType emote )
```

Here is the call graph for this function:



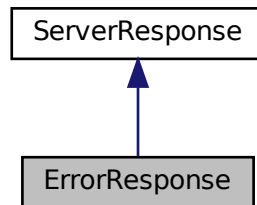
The documentation for this class was generated from the following files:

- `/home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.h`
- `/home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.cpp`

4.13 ErrorResponse Class Reference

```
#include <ErrorResponse.h>
```

Inheritance diagram for ErrorResponse:



Public Member Functions

- [ErrorResponse](#) ([BattleshipException](#) exception)

Public Attributes

- const [BattleshipException](#) exception

Additional Inherited Members

4.13.1 Constructor & Destructor Documentation

4.13.1.1 ErrorResponse()

```
ErrorResponse::ErrorResponse (
    BattleshipException exception ) [explicit]
```

4.13.2 Member Data Documentation

4.13.2.1 exception

```
const BattleshipException ErrorResponse::exception
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/ErrorResponse.h
- /home/nico/Desktop/battleship/src/common/network/responses/ErrorResponse.cpp

4.14 GameController Class Reference

```
#include <GameController.h>
```

Static Public Member Functions

- static void `init` (`GameWindow` *gameWindow)
Initializes all panels, displays the connection panel. Is also used to reset everything on GameOver or QuitGameEvent.
- static void `connectToServer` ()
Connects to server. Is called when connect button on ConnectionPanel is clicked.
- static void `enterSetupPhase` ()
enter setup phase. Is called when server responds with JoinGameSuccess response. Will show setup panel.
- static void `startGame` (const `StartGameSuccess` &response)
Function that is called when server responds with StartGameSuccess. Will show main game panel.
- static void `handleGameEvent` (const `GameEvent` &event)
Handles an incoming GameEvent. GameEvents are sent by the server when one of the players has placed a shot. The function will update the game state and display the new game state in the main game panel.
- static void `callShot` (`Coordinate` position)
Sends a shot request to the server.
- static void `sendEmote` (`EmoteType` emote)
Sends an emote to the server.
- static void `showEmote` (const `EmoteEvent` &emoteEvent)
Displays an emote to the screen and plays the corresponding sound.
- static void `showError` (const std::string &title, const std::string &message, bool popup)
Prints an error message to the console. If popup is true, it will also display a popup with the error message.
- static void `gameOver` (`uuid` winnerId)
Displays dialog box when game is finished and resets the game so both players are back at the connection panel.
- static void `handleQuitGameEvent` (`uuid` quitterId)
Shows popup saying that the other player left, closing the popup brings you back to the connection panel.
- static `SetupPanel` * `getSetupPanel` ()
- static wxEvtHandler * `getMainThreadEventHandler` ()
- static void `playerReady` ()
function that is called when ready button in SetupPanel is clicked. Will send request to server to start game and creates GameState used on client side.
- static void `quitGame` ()
Sends a QuitGame request to the server, is called upon closing the main window.

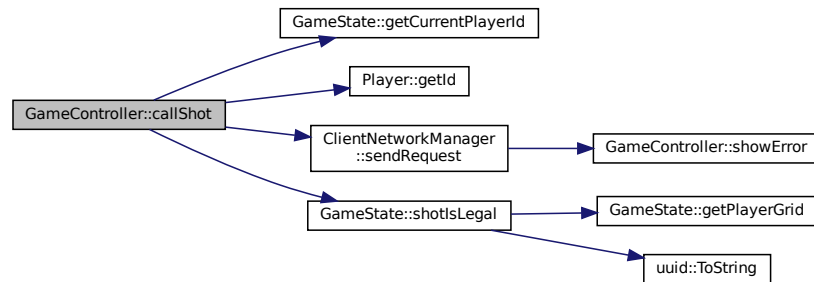
4.14.1 Member Function Documentation

4.14.1.1 callShot()

```
void GameController::callShot (
    Coordinate position ) [static]
```

Sends a shot request to the server.

Here is the call graph for this function:



Here is the caller graph for this function:

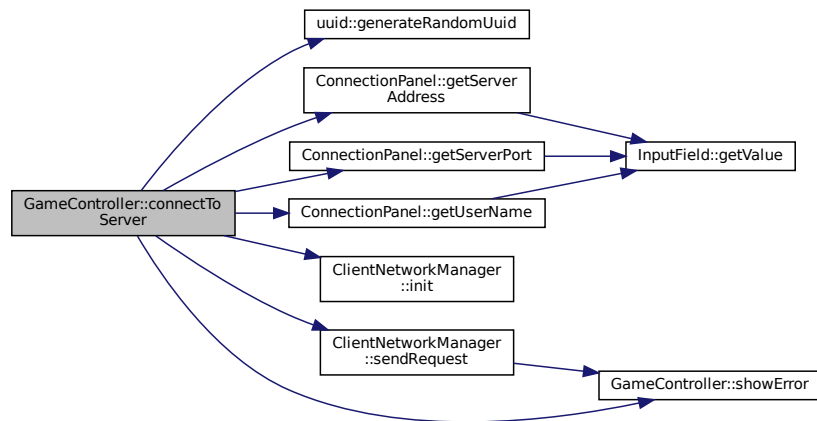


4.14.1.2 connectToServer()

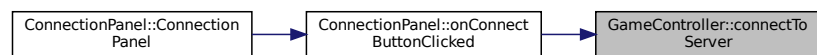
```
void GameController::connectToServer ( ) [static]
```

Connects to server. Is called when connect button on [ConnectionPanel](#) is clicked.

Here is the call graph for this function:



Here is the caller graph for this function:

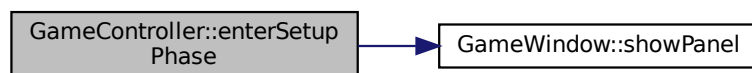


4.14.1.3 enterSetupPhase()

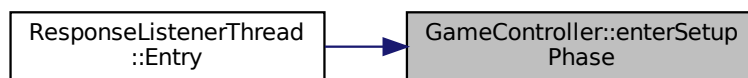
```
void GameController::enterSetupPhase ( ) [static]
```

enter setup phase. Is called when server responds with [JoinGameSuccess](#) response. Will show setup panel.

Here is the call graph for this function:



Here is the caller graph for this function:

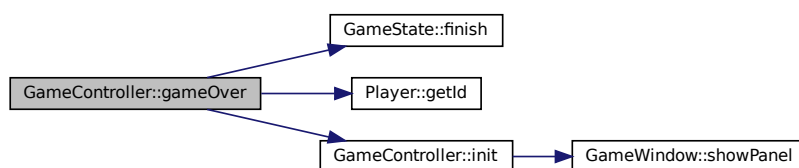


4.14.1.4 gameOver()

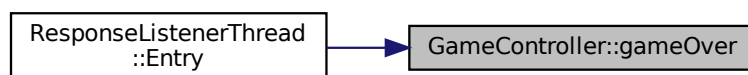
```
void GameController::gameOver (
    uuid winnerId ) [static]
```

Displays dialog box when game is finished and resets the game so both players are back at the connection panel.

Here is the call graph for this function:



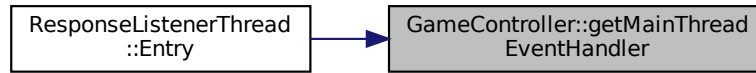
Here is the caller graph for this function:



4.14.1.5 getMainThreadEventHandler()

```
wxEvtHandler * GameController::getMainThreadEventHandler ( ) [static]
```

Here is the caller graph for this function:



4.14.1.6 getSetupPanel()

```
static SetupPanel* GameController::getSetupPanel ( ) [inline], [static]
```

Here is the caller graph for this function:

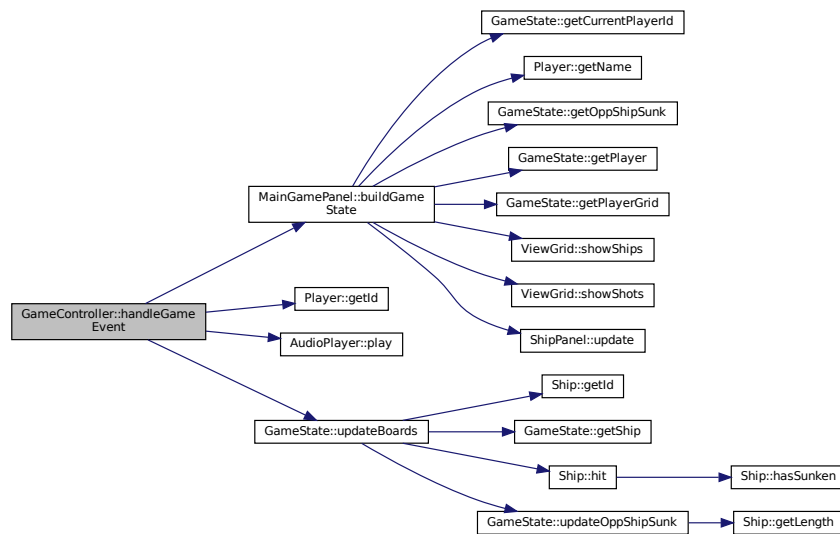


4.14.1.7 handleGameEvent()

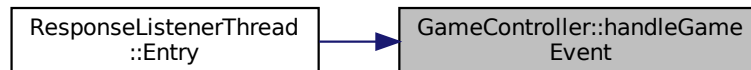
```
void GameController::handleGameEvent (
    const GameEvent & event ) [static]
```

Handles an incoming [GameEvent](#). GameEvents are sent by the server when one of the players has placed a shot. The function will update the game state and display the new game state in the main game panel.

Here is the call graph for this function:



Here is the caller graph for this function:



4.14.1.8 handleQuitGameEvent()

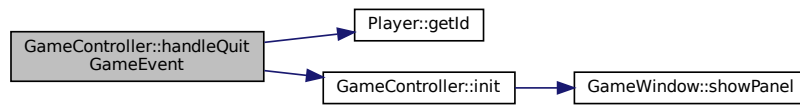
```
void GameController::handleQuitGameEvent (
    uuid quitterId ) [static]
```

Shows popup saying that the other player left, closing the popup brings you back to the connection panel.

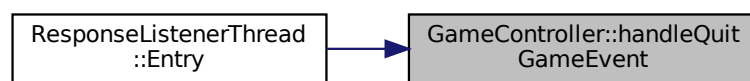
Parameters

<i>leaverId</i>	id of the player who quit
-----------------	---------------------------

Here is the call graph for this function:



Here is the caller graph for this function:



4.14.1.9 init()

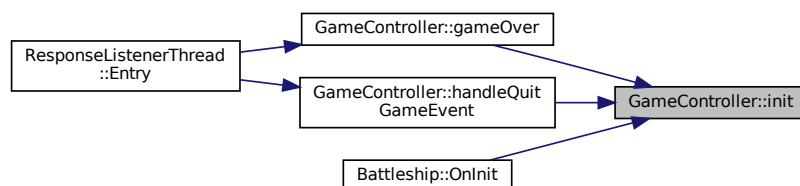
```
void GameController::init (
    GameWindow * gameWindow ) [static]
```

Initializes all panels, displays the connection panel. Is also used to reset everything on `GameOver` or `QuitGameEvent`.

Here is the call graph for this function:



Here is the caller graph for this function:

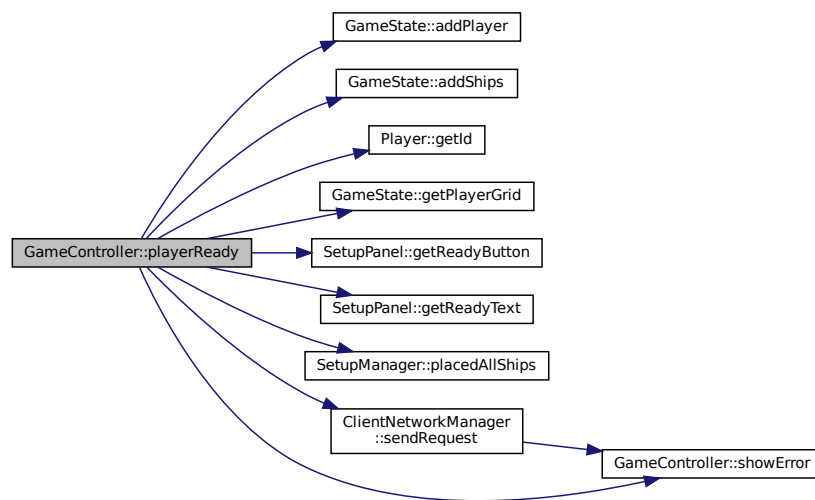


4.14.1.10 playerReady()

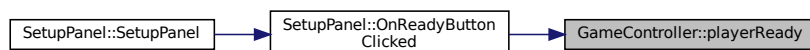
```
void GameController::playerReady ( ) [static]
```

function that is called when ready button in [SetupPanel](#) is clicked. Will send request to server to start game and creates [GameState](#) used on client side.

Here is the call graph for this function:



Here is the caller graph for this function:

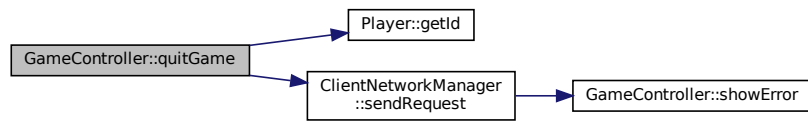


4.14.1.11 quitGame()

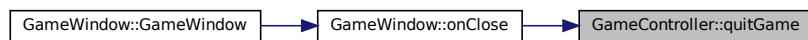
```
void GameController::quitGame ( ) [static]
```

Sends a [QuitGame](#) request to the server, is called upon closing the main window.

Here is the call graph for this function:



Here is the caller graph for this function:



4.14.1.12 sendEmote()

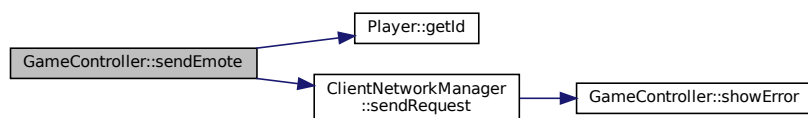
```
void GameController::sendEmote (
    EmoteType emote ) [static]
```

Sends an emote to the server.

Parameters

<i>emote</i>	The emote to be sent. EmoteType is an enum with 6 possible emotes.
--------------	--

Here is the call graph for this function:



Here is the caller graph for this function:

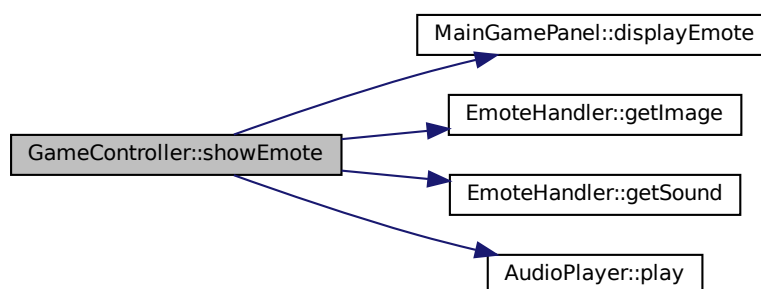


4.14.1.13 showEmote()

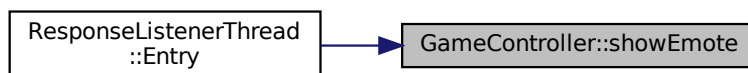
```
void GameController::showEmote (
    const EmoteEvent & emoteEvent ) [static]
```

Displays an emote to the screen and plays the corresponding sound.

Here is the call graph for this function:



Here is the caller graph for this function:



4.14.1.14 showError()

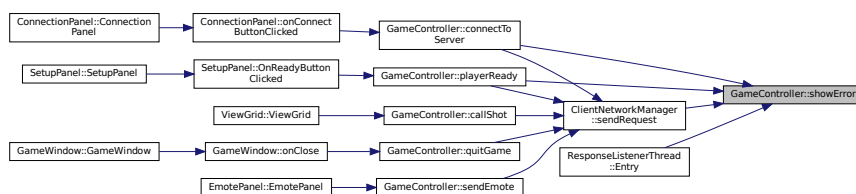
```
void GameController::showError (
    const std::string & title,
    const std::string & message,
    bool popup ) [static]
```

Prints an error message to the console. If popup is true, it will also display a popup with the error message.

Parameters

<i>title</i>	The title of the popup.
<i>message</i>	The message to be displayed.
<i>popup</i>	Whether or not to display a popup window.

Here is the caller graph for this function:

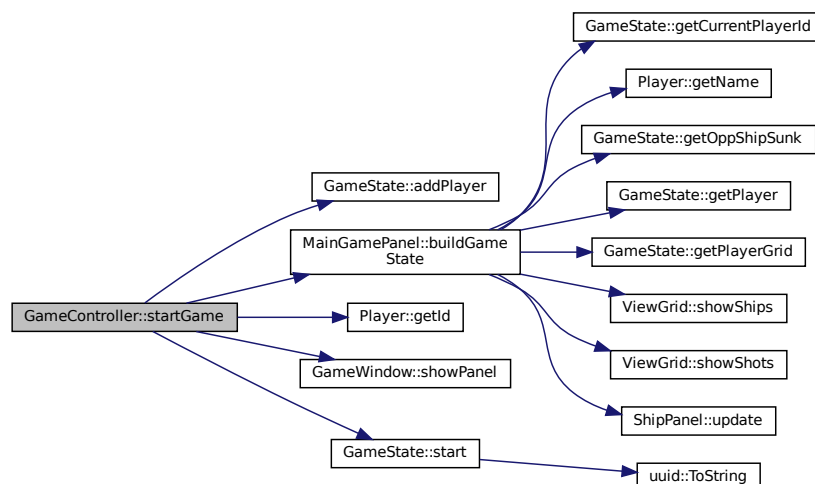


4.14.1.15 startGame()

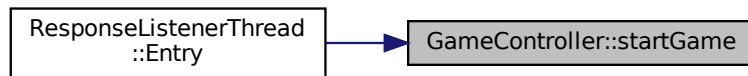
```
void GameController::startGame (
    const StartGameSuccess & response ) [static]
```

Function that is called when server responds with [StartGameSuccess](#). Will show main game panel.

Here is the call graph for this function:



Here is the caller graph for this function:



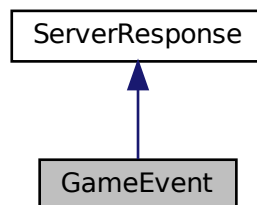
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/GameController.h](#)
- [/home/nico/Desktop/battleship/src/client/GameController.cpp](#)

4.15 GameEvent Class Reference

```
#include <GameEvent.h>
```

Inheritance diagram for GameEvent:



Public Member Functions

- [GameEvent](#) ([uuid playerId](#), [Coordinate position](#), [bool hit](#), [bool sunk](#), [Ship hitShip](#), [uuid nextPlayerId](#))

Public Attributes

- [const uuid playerId](#)
- [const Coordinate position](#)
- [const bool hit](#) = false
- [const bool sunk](#) = false
- [const Ship hitShip](#)
- [const uuid nextPlayerId](#)

Additional Inherited Members

4.15.1 Detailed Description

A Game Event is emitted by the server if and only if a new shot was registered

4.15.2 Constructor & Destructor Documentation

4.15.2.1 GameEvent()

```
GameEvent::GameEvent (
    uuid playerId,
    Coordinate position,
    bool hit,
    bool sunk,
    Ship hitShip,
    uuid nextPlayerId )
```

4.15.3 Member Data Documentation

4.15.3.1 hit

```
const bool GameEvent::hit = false
```

4.15.3.2 hitShip

```
const Ship GameEvent::hitShip
```

4.15.3.3 nextPlayerId

```
const uuid GameEvent::nextPlayerId
```

4.15.3.4 playerId

```
const uuid GameEvent::playerId
```

4.15.3.5 position

```
const Coordinate GameEvent::position
```

4.15.3.6 sunk

```
const bool GameEvent::sunk = false
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/[GameEvent.h](#)
- /home/nico/Desktop/battleship/src/common/network/responses/[GameEvent.cpp](#)

4.16 GameInstance Class Reference

```
#include <GameInstance.h>
```

Public Member Functions

- bool [joinGame](#) (const [JoinGame](#) &joinGameRequest)
- bool [startGame](#) (const [Player](#) &player, std::string &err)
- bool [executeShot](#) (const [CallShot](#) &shotRequest)
- bool [reset](#) ()
- [GameState](#) & [getGameState](#) ()
- bool [isReady](#) (const [Player](#) &player)

4.16.1 Member Function Documentation

4.16.1.1 executeShot()

```
bool GameInstance::executeShot (  
    const CallShot & shotRequest )
```

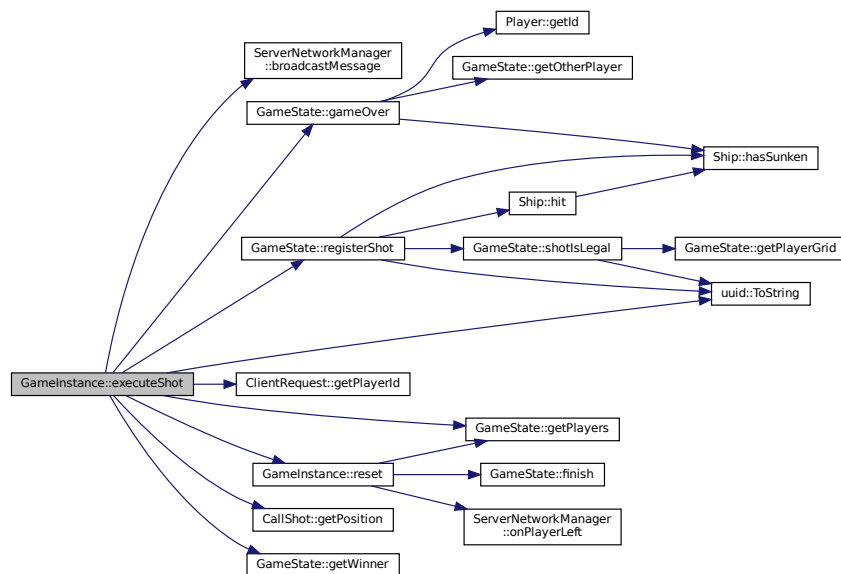
handles a call shot request. registers the shot in the gamestate and sends out a corresponding gameEvent

Parameters

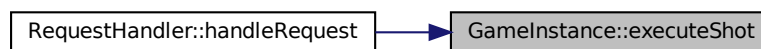
<i>shotRequest</i>	
--------------------	--

Returns

Here is the call graph for this function:



Here is the caller graph for this function:



4.16.1.2 getGameState()

`GameState` & `GameInstance::getGameState ()`

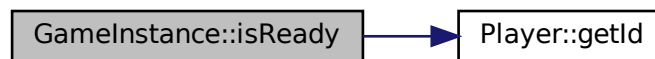
Here is the caller graph for this function:



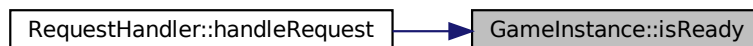
4.16.1.3 isReady()

```
bool GameInstance::isReady (
    const Player & player )
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.16.1.4 joinGame()

```
bool GameInstance::joinGame (
    const JoinGame & joinGameRequest )
```

handles a client request to join the game. adds player to the [GameState](#)

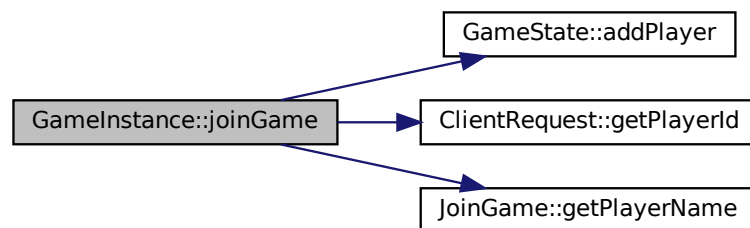
Parameters

<i>joinGameRequest</i>	contains player name and id
------------------------	-----------------------------

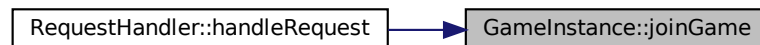
Returns

true if player was added successfully, false otherwise

Here is the call graph for this function:



Here is the caller graph for this function:



4.16.1.5 reset()

```
bool GameInstance::reset ( )
```

Function to handle reset after [QuitGame](#) or `GameOver`. Recreates a new [GameState](#) to be ready for next Game. Also removes current players network information from the [ServerNetworkManager](#) <>

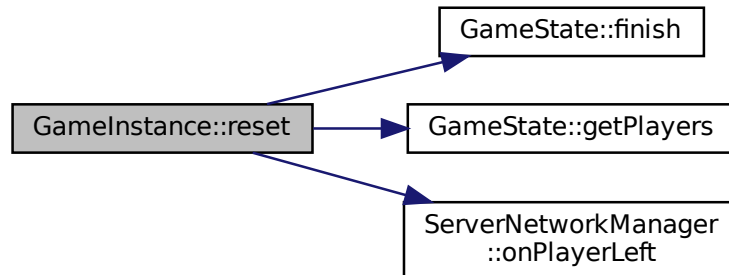
Parameters

QuitGame	
--------------------------	--

Returns

True if all `_gameState` attributes are reset. Else, false.

Here is the call graph for this function:



Here is the caller graph for this function:

**4.16.1.6 startGame()**

```

bool GameInstance::startGame (
    const Player & player,
    std::string & err )
  
```

starts the game as soon as both players are ready

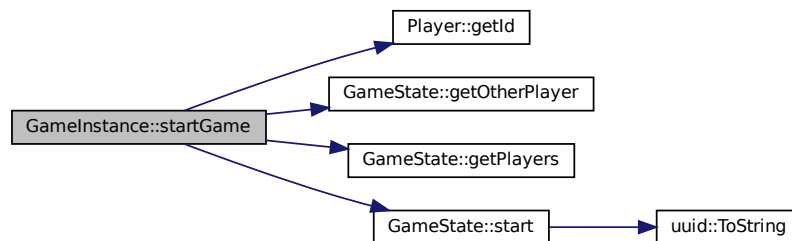
Parameters

<i>player</i>	the player who pressed ready
<i>err</i>	

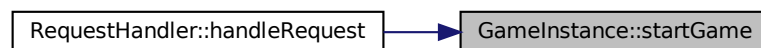
Returns

true if game was started. false if not started

Here is the call graph for this function:



Here is the caller graph for this function:



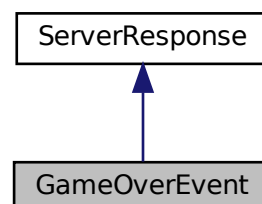
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/server/GameInstance.h](#)
- [/home/nico/Desktop/battleship/src/server/GameInstance.cpp](#)

4.17 GameOverEvent Class Reference

```
#include <GameOverEvent.h>
```

Inheritance diagram for GameOverEvent:



Public Member Functions

- [GameOverEvent](#) ([uuid](#) [winnerPlayerId](#))

Public Attributes

- const [uuid](#) [winnerPlayerId](#)

Additional Inherited Members

4.17.1 Constructor & Destructor Documentation

4.17.1.1 GameOverEvent()

```
GameOverEvent::GameOverEvent (
    uuid winnerPlayerId ) [explicit]
```

4.17.2 Member Data Documentation

4.17.2.1 winnerPlayerId

```
const uuid GameOverEvent::winnerPlayerId
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/[GameOverEvent.h](#)
- /home/nico/Desktop/battleship/src/common/network/responses/[GameOverEvent.cpp](#)

4.18 GameState Class Reference

```
#include <GameState.h>
```

Public Types

- enum class [State](#) { [Starting](#) , [Playing](#) , [Finished](#) }
- enum class [Type](#) { [ServerState](#) , [ClientState](#) }

Public Member Functions

- [GameState](#) ([GameState::Type](#) type)
- bool [addPlayer](#) ([Player](#) player)
- bool [removePlayer](#) ([Player](#) player)
- bool [addShips](#) (uuid playerId, std::vector< [Ship](#) > shipPlacement)
- bool [start](#) (uuid currentPlayerId)
- uuid [getCurrentPlayerId](#) ()
- const [PlayerGrid](#) & [getPlayerGrid](#) (uuid playerId) const
- const std::vector< [Player](#) > & [getPlayers](#) () const
- [Ship](#) * [getShip](#) (std::vector< [Ship](#) > &ships, uuid shipId)
- const [Player](#) * [getPlayer](#) (uuid playerId) const
- const [Player](#) * [getOtherPlayer](#) (uuid playerId)
- const [State](#) & [getState](#) () const
- const std::array< bool, 5 > & [getOppShipSunk](#) ()
- bool [shotsLegal](#) (uuid playerId, [Coordinate](#) position)
- bool [registerShot](#) (uuid playerId, [Coordinate](#) position, bool *hit, [Ship](#) **hitShipPtr, bool *sunk, uuid *nextPlayerId)
- bool [updateBoards](#) (const [GameEvent](#) &event)
updates the CLIENT SIDE boards after a game event happened.
- bool [updateOppShipSunk](#) (const [Ship](#) &hitShip)
- bool [gameOver](#) ()
- uuid [getWinner](#) ()
- void [finish](#) ()

4.18.1 Member Enumeration Documentation

4.18.1.1 State

```
enum GameState::State [strong]
```

Enum indicating the phase of the game state. Must be switched to Playing before first move is made

Enumerator

Starting	
Playing	
Finished	

4.18.1.2 Type

```
enum GameState::Type [strong]
```

Enum indicating whether a server side game state is being stored or a client side game state

Enumerator

ServerState	
ClientState	

4.18.2 Constructor & Destructor Documentation

4.18.2.1 GameState()

```
GameState::GameState (
    GameState::Type type )
```

Constructor setting initial values

Parameters

<i>type</i>	defines if this is a server side or client side state
-------------	---

4.18.3 Member Function Documentation

4.18.3.1 addPlayer()

```
auto GameState::addPlayer (
    Player player )
```

Adds a player to the game state

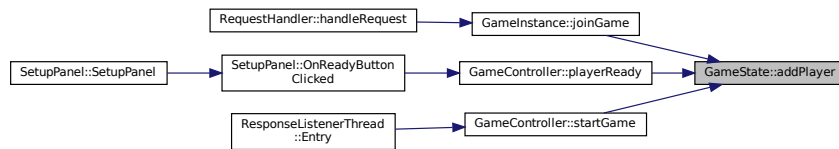
Parameters

<i>player</i>	the player object
---------------	-------------------

Returns

true if player was added successfully, false if a problem occurred.

Here is the caller graph for this function:

**4.18.3.2 addShips()**

```

auto GameState::addShips (
    uuid playerId,
    std::vector< Ship > shipPlacement )

```

Creates a new playerGrid from a full ship placement

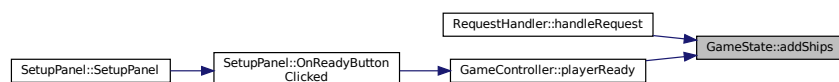
Parameters

<i>playerId</i>	id of the player who placed the ships and owns the board
<i>shipPlacement</i>	a vector containing all ships

Returns

true if grid was created successfully, false if a problem occurred.

Here is the caller graph for this function:

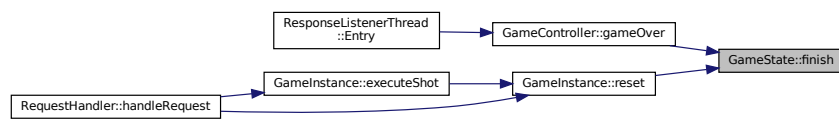
**4.18.3.3 finish()**

```

void GameState::finish ( )

```

sets gamestate to Finished. For a next game, a new gamestate should be created. Thus this does not reset everything. Here is the caller graph for this function:



4.18.3.4 gameOver()

```
bool GameState::gameOver ( )
```

Checks if the game is over meaning all ships of one player are sunk. Only to be called by the server.

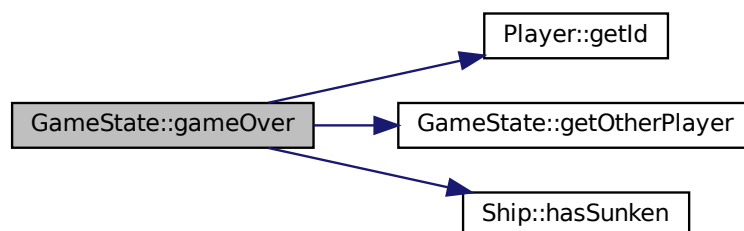
Parameters

<i>winner</i>	id of the winning player. will be "0" if game is not over.
---------------	--

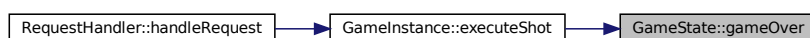
Returns

true if game is over. false if game is not over

Here is the call graph for this function:



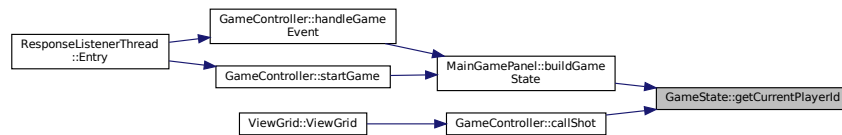
Here is the caller graph for this function:



4.18.3.5 getCurrentPlayerId()

```
uuid GameState::getCurrentPlayerId ( )
```

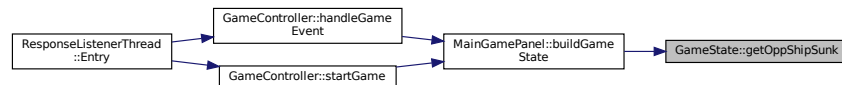
getter for the id of the player whose turn it currently is Here is the caller graph for this function:



4.18.3.6 getOppShipSunk()

```
const std::array< bool, 5 > & GameState::getOppShipSunk ( )
```

returns an array of bool which indicates which opponent ships were already sunk. used for crossing out ships at the bottom of the UI Here is the caller graph for this function:



4.18.3.7 getOtherPlayer()

```
const Player * GameState::getOtherPlayer (
    uuid playerId )
```

returns the id of the player who has NOT the id specified as parameter

Parameters

<i>player</i> ↔ <i>Id</i>	
------------------------------	--

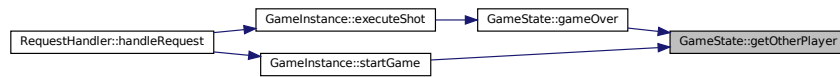
Returns

id of the other player. "Error" if no other player found.

Precondition

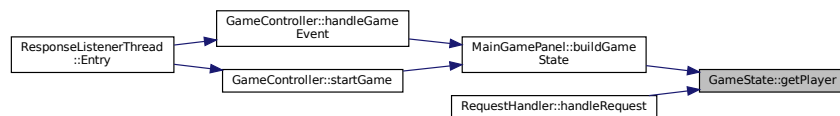
there are exactly 2 players added to the gameState

Here is the caller graph for this function:

**4.18.3.8 getPlayer()**

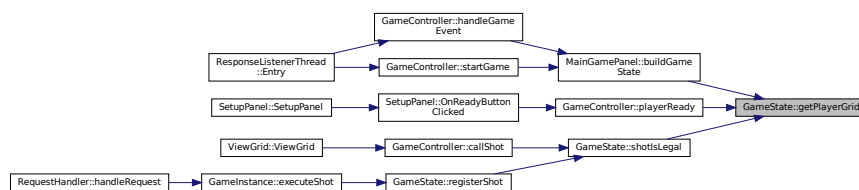
```
const Player * GameState::getPlayer (
    uuid playerId ) const
```

returns the name of the player with the specified id Here is the caller graph for this function:

**4.18.3.9 getPlayerGrid()**

```
const PlayerGrid & GameState::getPlayerGrid (
    uuid playerId ) const
```

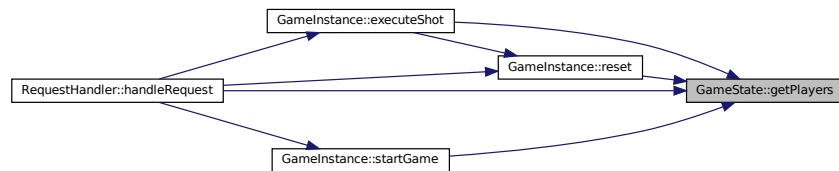
returns a const reference of the grid for e.g. display through UI Here is the caller graph for this function:



4.18.3.10 getPlayer()

```
const std::vector< Player > & GameState::getPlayers ( ) const
```

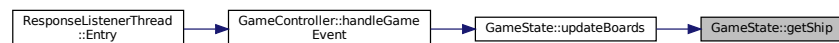
returns a vector of all the players (2) Here is the caller graph for this function:



4.18.3.11 getShip()

```
Ship * GameState::getShip (
    std::vector< Ship > & ships,
    uuid shipId )
```

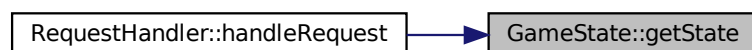
returns a reference to the ship with the specified id from a given vector of ships Here is the caller graph for this function:



4.18.3.12 getState()

```
const GameState::State & GameState::getState ( ) const
```

returns the current state of the game (starting, playing, finished) Here is the caller graph for this function:



4.18.3.13 `getWinner()`

```
uuid GameState::getWinner ( )
```

Returns the winner if the game is over. Should only be called after `gameOver()` returned true.

Returns

id of winner. null-uuid if game is not over

Here is the caller graph for this function:



4.18.3.14 `registerShot()`

```
bool GameState::registerShot (
    uuid playerId,
    Coordinate position,
    bool * hit,
    Ship ** hitShipPtr,
    bool * sunk,
    uuid * nextPlayerId )
```

Function to register a shot ON THE SERVER SIDE and process the results

All arguments from *hit on are to be used to return info back to the caller (in this case to gameInstance) Warning: The double pointer for the ship is intentional. Don't change please!

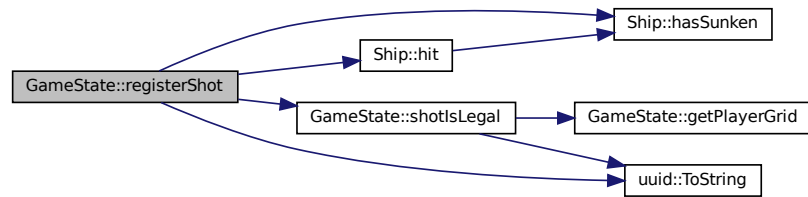
Parameters

<i>playerId</i>	ID of the player who called the shot
<i>position</i>	location of the shot
<i>hit</i>	holds info on whether the shot was a hit
<i>hitShip</i>	holds info on the (potentially) hit ship for the caller to extract (and emit a GameEvent) Turns the pointer to nullptr if no hit is false.
<i>sunk</i>	holds info on whether the hit ship has sunk
<i>nextPlayerId</i>	holds id of the player who has to play next

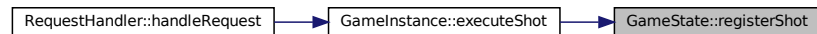
Returns

true if the shot was registered properly, false if a problem occurred

Here is the call graph for this function:



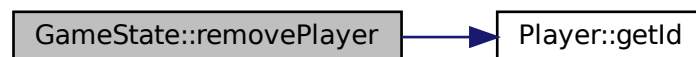
Here is the caller graph for this function:



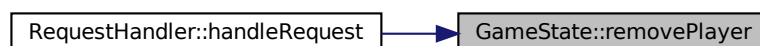
4.18.3.15 removePlayer()

```
bool GameState::removePlayer (
    Player player )
```

Removes a player from the game state if state is still in setup phase. If a player leaves while playing the game ends and a new gameState has to be created. Here is the call graph for this function:



Here is the caller graph for this function:



4.18.3.16 shotIsLegal()

```
bool GameState::shotIsLegal (
    uuid playerId,
    Coordinate position )
```

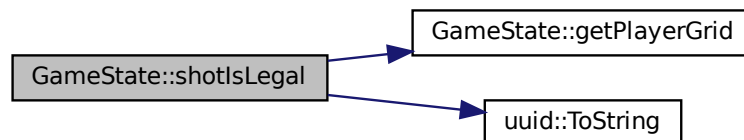
checks if a called shot is a legal move

Parameters

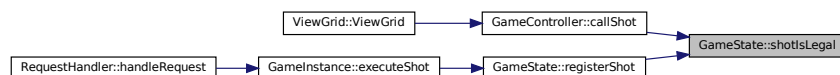
<i>playerId</i>	ID of the player calling the shot
<i>position</i>	location of the shot

Returns

Here is the call graph for this function:



Here is the caller graph for this function:



4.18.3.17 start()

```
bool GameState::start (
    uuid currentPlayerId )
```

Switches the game state from `State::Starting` to `State::Playing`

Precondition

Two players were added to the state. Grids were added (1 for clients, 2 for servers).

Parameters

<i>current</i> ↔ <i>PlayerId</i>	id of the player who will go first
-------------------------------------	------------------------------------

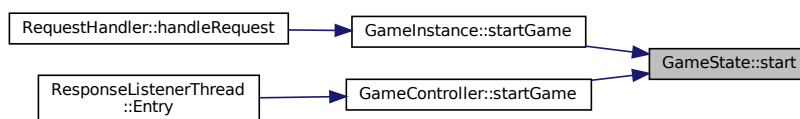
Returns

true if the game was started, false if a problem occurred. Check log file for details.

Here is the call graph for this function:



Here is the caller graph for this function:



4.18.3.18 updateBoards()

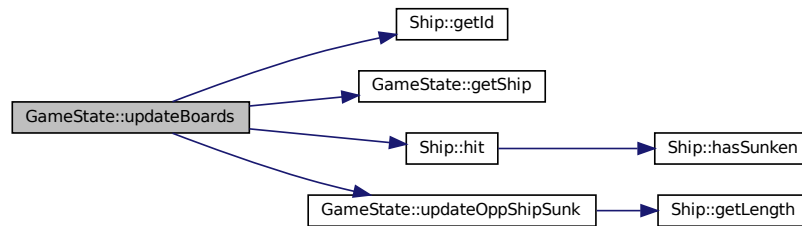
```
bool GameState::updateBoards (
    const GameEvent & event )
```

updates the CLIENT SIDE boards after a game event happened.

Parameters

<i>event</i>	
--------------	--

Here is the call graph for this function:



Here is the caller graph for this function:



4.18.3.19 updateOppShipSunk()

```
bool GameState::updateOppShipSunk (
    const Ship & hitShip )
```

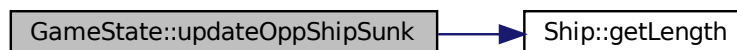
updates the array to cross out ships on the bottom of the screen

Parameters

<i>hitShip</i>	
----------------	--

Returns

Here is the call graph for this function:



Here is the caller graph for this function:



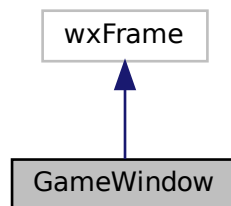
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/game_state/GameState.h](#)
- [/home/nico/Desktop/battleship/src/common/game_state/GameState.cpp](#)

4.19 GameWindow Class Reference

```
#include <GameWindow.h>
```

Inheritance diagram for GameWindow:



Public Member Functions

- [GameWindow](#) (const wxString &title, const wxPoint &pos, const wxSize &size)
- void [showPanel](#) (wxPanel *panel)
- void [setStatus](#) (const std::string &message)
- void [onClose](#) (wxCloseEvent &event)

4.19.1 Constructor & Destructor Documentation

4.19.1.1 `GameWindow()`

```
GameWindow::GameWindow (
    const wxString & title,
    const wxPoint & pos,
    const wxSize & size )
```

Here is the call graph for this function:

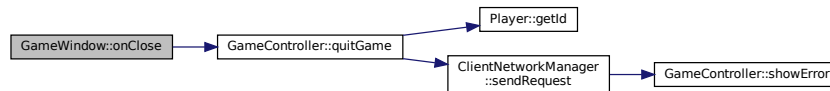


4.19.2 Member Function Documentation

4.19.2.1 `onClose()`

```
void GameWindow::onClose (
    wxCloseEvent & event )
```

Here is the call graph for this function:



Here is the caller graph for this function:



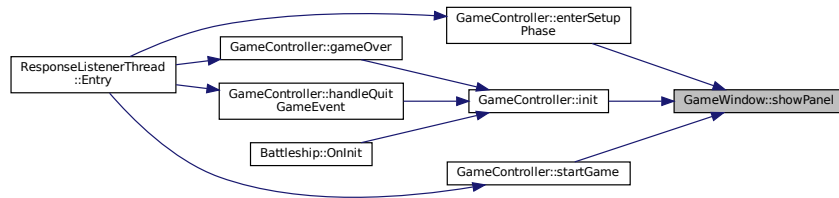
4.19.2.2 `setStatus()`

```
void GameWindow::setStatus (
    const std::string & message )
```

4.19.2.3 showPanel()

```
void GameWindow::showPanel (
    wxPanel * panel )
```

Here is the caller graph for this function:



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

- </home/nico/Desktop/battleship/src/client/GameWindow.h>
- </home/nico/Desktop/battleship/src/client/GameWindow.cpp>

4.20 std::hash< uuid > Struct Reference

```
#include <uuid.h>
```

Public Member Functions

- `std::size_t operator() (uuid const &id) const` noexcept

4.20.1 Member Function Documentation

4.20.1.1 operator()()

```
std::size_t std::hash< uuid >::operator() (
    uuid const & id ) const [inline], [noexcept]
```

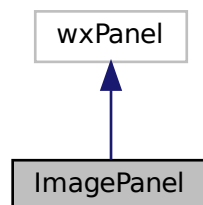
The documentation for this struct was generated from the following file:

- </home/nico/Desktop/battleship/src/common/uuid.h>

4.21 ImagePanel Class Reference

```
#include <ImagePanel.h>
```

Inheritance diagram for ImagePanel:



Public Member Functions

- [ImagePanel](#) (`wxWindow *parent`, `wxString file`, `wxBitmapType format`, `wxPoint position=wxDefaultPosition`, `wxSize size=wxDefaultSize`, `double rotation=0.0`)
- void [paintEvent](#) (`wxPaintEvent &event`)
- void [onSize](#) (`wxSizeEvent &event`)

4.21.1 Detailed Description

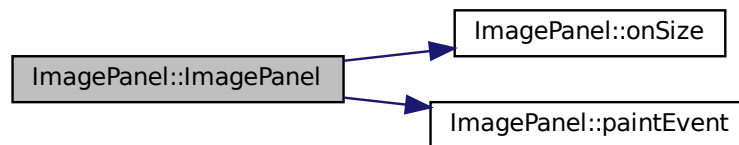
This class can be used to display an image. It can be scaled with parameter `<size>` and rotated with `<rotation>` (in radian)

4.21.2 Constructor & Destructor Documentation

4.21.2.1 ImagePanel()

```
ImagePanel::ImagePanel (  
    wxWindow * parent,  
    wxString file,  
    wxBitmapType format,  
    wxPoint position = wxDefaultPosition,  
    wxSize size = wxDefaultSize,  
    double rotation = 0.0 )
```

Here is the call graph for this function:



4.21.3 Member Function Documentation

4.21.3.1 onSize()

```
void ImagePanel::onSize (
    wxSizeEvent & event )
```

Here is the caller graph for this function:



4.21.3.2 paintEvent()

```
void ImagePanel::paintEvent (
    wxPaintEvent & event )
```

Here is the caller graph for this function:



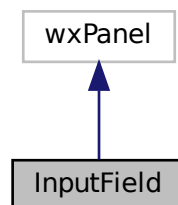
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.h](#)
- [/home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.cpp](#)

4.22 InputField Class Reference

```
#include <InputField.h>
```

Inheritance diagram for InputField:



Public Member Functions

- [InputField](#) (wxWindow *parent, const wxString &labelText, int labelWidth, const wxString &fieldValue, int fieldWidth)
- wxString [getValue](#) ()

4.22.1 Constructor & Destructor Documentation

4.22.1.1 InputField()

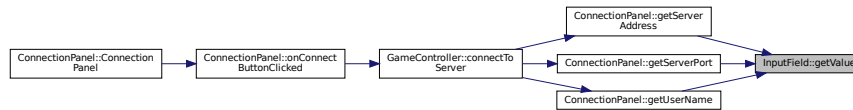
```
InputField::InputField (  
    wxWindow * parent,  
    const wxString & labelText,  
    int labelWidth,  
    const wxString & fieldValue,  
    int fieldWidth )
```

4.22.2 Member Function Documentation

4.22.2.1 `getValue()`

```
wxString InputField::getValue ( )
```

Here is the caller graph for this function:



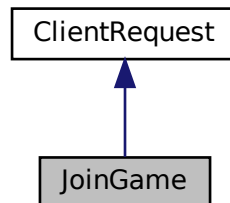
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/uiElements/InputField.h](#)
- [/home/nico/Desktop/battleship/src/client/uiElements/InputField.cpp](#)

4.23 JoinGame Class Reference

```
#include <JoinGame.h>
```

Inheritance diagram for JoinGame:



Public Member Functions

- [JoinGame](#) ([uuid](#) playerId, [std::string](#) playerName)
- [auto](#) [getPlayerName](#) () [const](#) -> [std::string](#)

Additional Inherited Members

4.23.1 Constructor & Destructor Documentation

4.23.1.1 JoinGame()

```
JoinGame::JoinGame (
    uuid playerId,
    std::string playerName )
```

4.23.2 Member Function Documentation

4.23.2.1 getPlayerName()

```
auto JoinGame::getPlayerName ( ) const -> std::string
```

Here is the caller graph for this function:



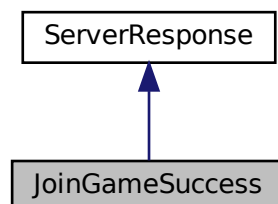
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/network/requests/JoinGame.h](#)
- [/home/nico/Desktop/battleship/src/common/network/requests/JoinGame.cpp](#)

4.24 JoinGameSuccess Class Reference

```
#include <JoinGameSuccess.h>
```

Inheritance diagram for JoinGameSuccess:



Public Member Functions

- [JoinGameSuccess](#) ()
- bool [wasSuccessful](#) () const

Additional Inherited Members

4.24.1 Detailed Description

[ServerResponse](#) to a [ClientRequest](#) of whether it was successful or not

4.24.2 Constructor & Destructor Documentation

4.24.2.1 JoinGameSuccess()

```
JoinGameSuccess::JoinGameSuccess ( )
```

4.24.3 Member Function Documentation

4.24.3.1 wasSuccessful()

```
bool JoinGameSuccess::wasSuccessful ( ) const
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/[JoinGameSuccess.h](#)
- /home/nico/Desktop/battleship/src/common/network/responses/[JoinGameSuccess.cpp](#)

4.25 Logger Class Reference

```
#include <Logger.h>
```

Static Public Member Functions

- static void [log](#) (const std::string &message, const std::string &function="-")
- static void [log](#) (const [BattleshipException](#) &exception)
- static void [setPrefix](#) (const std::string &s)

4.25.1 Detailed Description

Custom basic logging class. Use the macro `LOG(msg)` anywhere in the project to log whatever you need.

4.25.2 Member Function Documentation

4.25.2.1 `log()` [1/2]

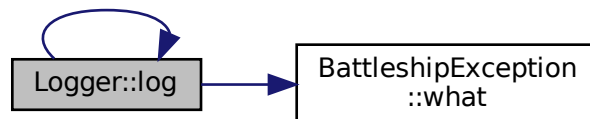
```
void Logger::log (  
    const BattleshipException & exception ) [static]
```

overloading of `Logger::log` to output BattleshipExceptions directly

Parameters

<i>exception</i>	
------------------	--

Here is the call graph for this function:



Here is the caller graph for this function:



4.25.2.2 log() [2/2]

```
void Logger::log (
    const std::string & message,
    const std::string & function = "-" ) [static]
```

outputs a message to `std::cout` and to the logfile. Use the [LOG\(msg\)](#) macro instead of this function whenever possible

Parameters

<i>message</i>	text to be logged
<i>function</i>	name of the function where this log was called. automatically filled in by the macro

4.25.2.3 setPrefix()

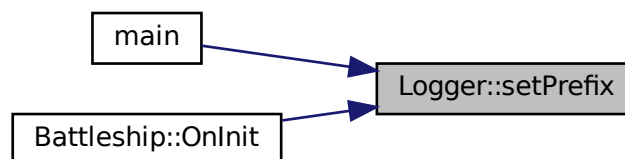
```
void Logger::setPrefix (
    const std::string & s ) [static]
```

changes the prefix used at the very beginning of a log message. use this if you run multiple instances (servers and clients) on the same machine to identify more easily where the message came from

Parameters

<i>s</i>	
----------	--

Here is the caller graph for this function:



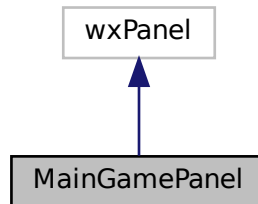
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/Logger.h](#)
- [/home/nico/Desktop/battleship/src/common/Logger.cpp](#)

4.26 MainGamePanel Class Reference

```
#include <MainGamePanel.h>
```

Inheritance diagram for MainGamePanel:



Public Member Functions

- `MainGamePanel` (`wxWindow *parent`)
Constructor for the main game panel. Doesn't display anything, only the background color.
- void `buildGameState` (`GameState *gameState`, `uuid ownId`)
Builds the game state.
- void `displayEmote` (`EmoteType emote`)
Displays the emote sent by the other player.

4.26.1 Constructor & Destructor Documentation

4.26.1.1 MainGamePanel()

```
MainGamePanel::MainGamePanel (  
    wxWindow * parent )
```

Constructor for the main game panel. Doesn't display anything, only the background color.

4.26.2 Member Function Documentation

4.26.2.1 buildGameState()

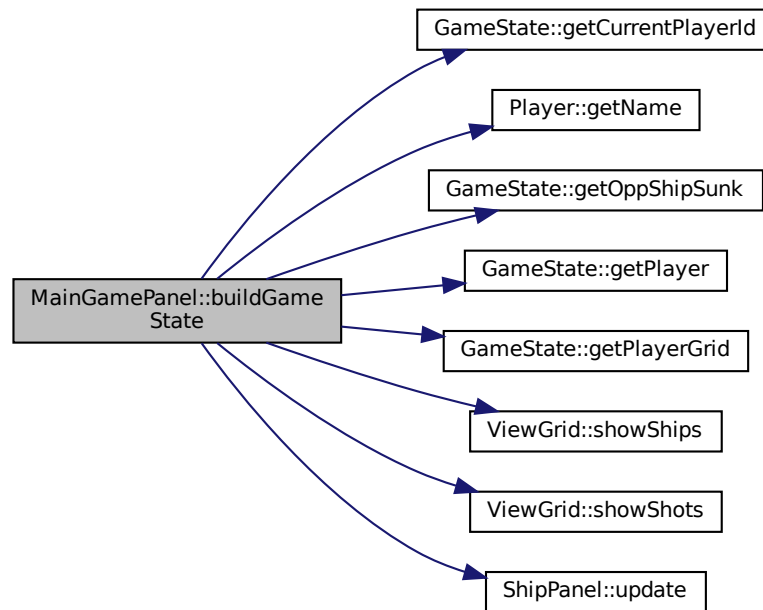
```
void MainGamePanel::buildGameState (  
    GameState * gameState,  
    uuid ownId )
```

Builds the game state.

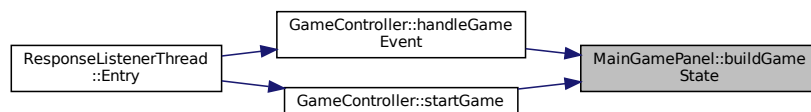
Parameters

<i>gameState</i>	The game state.
<i>ownId</i>	The id of the player who is using this client. Displays the game state using ViewGrid and ShipPanel .

Here is the call graph for this function:



Here is the caller graph for this function:



4.26.2.2 displayEmote()

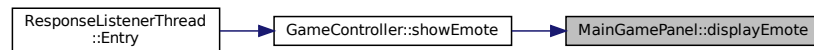
```
void MainGamePanel::displayEmote (
    EmoteType emote )
```

Displays the emote sent by the other player.

Parameters

<i>emote</i>	The emote type that was sent. Not implemented yet.
--------------	--

Here is the caller graph for this function:



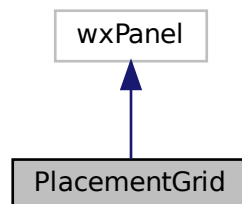
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/panels/MainGamePanel.h](#)
- [/home/nico/Desktop/battleship/src/client/panels/MainGamePanel.cpp](#)

4.27 PlacementGrid Class Reference

```
#include <PlacementGrid.h>
```

Inheritance diagram for PlacementGrid:



Public Member Functions

- [PlacementGrid](#) (wxWindow *parent)
constructor for [PlacementGrid](#). Creates grid of 10x10 tiles and binds mouse events
- void [OnMouseMotion](#) (wxMouseEvent &event)
function is called when mouse hovers over grid and highlights tiles according to mouse position and placed ships
- void [OnMouseClicked](#) (wxMouseEvent &event)
function is called when mouse clicks on grid and places ship if possible
- void [displayGrid](#) ()
function displays grid according to data in [SetupManager](#)
- void [highlightTiles](#) (int CellX, int CellY)
function highlights tiles according to mouse position and placed ships

Public Attributes

- int [cellX_prev](#) = -1
- int [cellY_prev](#) = -1

4.27.1 Constructor & Destructor Documentation

4.27.1.1 PlacementGrid()

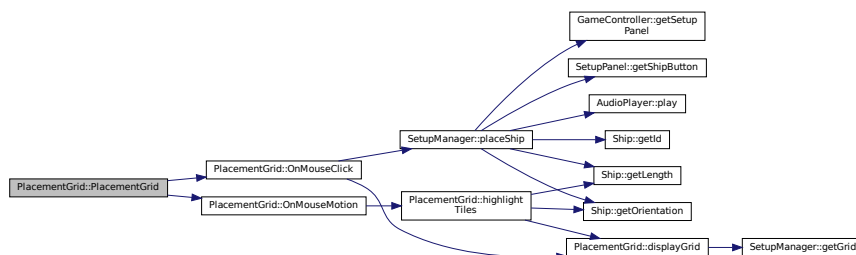
```
PlacementGrid::PlacementGrid (
    wxWindow * parent )
```

constructor for [PlacementGrid](#). Creates grid of 10x10 tiles and binds mouse events

Parameters

<i>parent</i>	
---------------	--

Here is the call graph for this function:



4.27.2 Member Function Documentation

4.27.2.1 displayGrid()

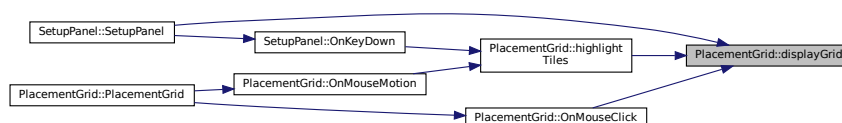
```
void PlacementGrid::displayGrid ( )
```

function displays grid according to data in [SetupManager](#)

Here is the call graph for this function:



Here is the caller graph for this function:



4.27.2.2 highlightTiles()

```

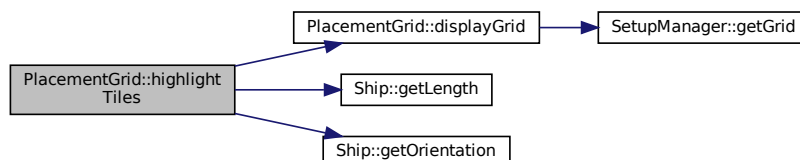
void PlacementGrid::highlightTiles (
    int CellX,
    int CellY )
  
```

function highlights tiles according to mouse position and placed ships

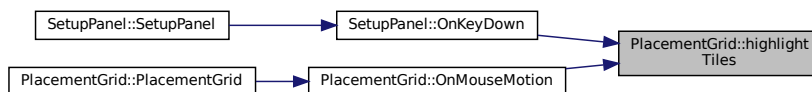
Parameters

<i>CellX</i>	x coordinate of mouse position
<i>CellY</i>	y coordinate of mouse position

Here is the call graph for this function:



Here is the caller graph for this function:



4.27.2.3 OnMouseClicked()

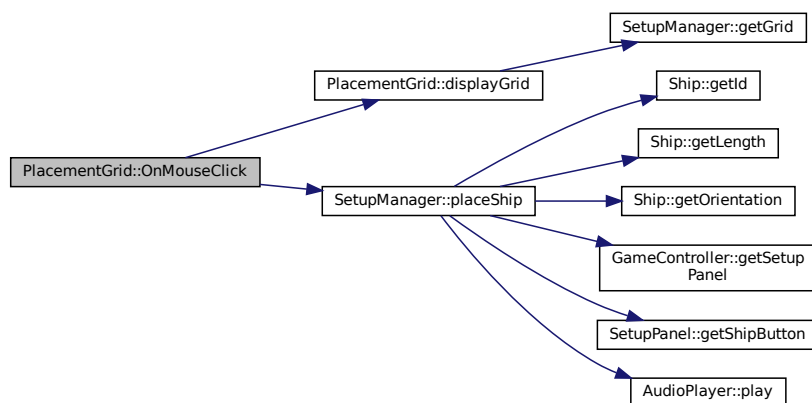
```
void PlacementGrid::OnMouseClicked (
    wxMouseEvent & event )
```

function is called when mouse clicks on grid and places ship if possible

Parameters

<i>event</i>	
--------------	--

Here is the call graph for this function:



Here is the caller graph for this function:



4.27.2.4 OnMouseMotion()

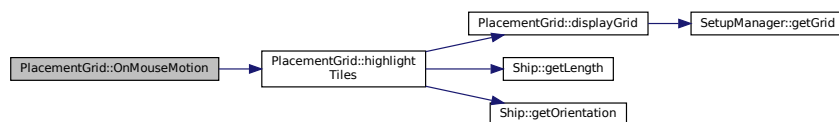
```
void PlacementGrid::OnMouseMotion (
    wxMouseEvent & event )
```

function is called when mouse hovers over grid and highlights tiles according to mouse position and placed ships

Parameters

<i>event</i>	mouse event
--------------	-------------

Here is the call graph for this function:



Here is the caller graph for this function:



4.27.3 Member Data Documentation

4.27.3.1 cellX_prev

```
int PlacementGrid::cellX_prev = -1
```

4.27.3.2 cellY_prev

```
int PlacementGrid::cellY_prev = -1
```

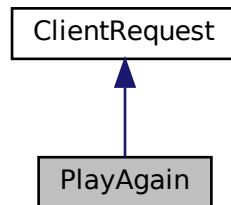
The documentation for this class was generated from the following files:

- </home/nico/Desktop/battleship/src/client/uiElements/PlacementGrid.h>
- </home/nico/Desktop/battleship/src/client/uiElements/PlacementGrid.cpp>

4.28 PlayAgain Class Reference

```
#include <PlayAgain.h>
```

Inheritance diagram for PlayAgain:



Public Member Functions

- [PlayAgain](#) ([uuid](#) playerId)

Additional Inherited Members

4.28.1 Constructor & Destructor Documentation

4.28.1.1 PlayAgain()

```
PlayAgain::PlayAgain (  
    uuid playerId ) [explicit]
```

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

- [/home/nico/Desktop/battleship/src/common/network/requests/PlayAgain.h](#)
- [/home/nico/Desktop/battleship/src/common/network/requests/PlayAgain.cpp](#)

4.29 Player Class Reference

```
#include <Player.h>
```

Public Member Functions

- `Player (uuid playerId, std::string player_name)`
- `auto getId () const -> uuid`
- `auto getName () const -> std::string`
- `bool operator== (const Player &) const =default`

4.29.1 Constructor & Destructor Documentation

4.29.1.1 Player()

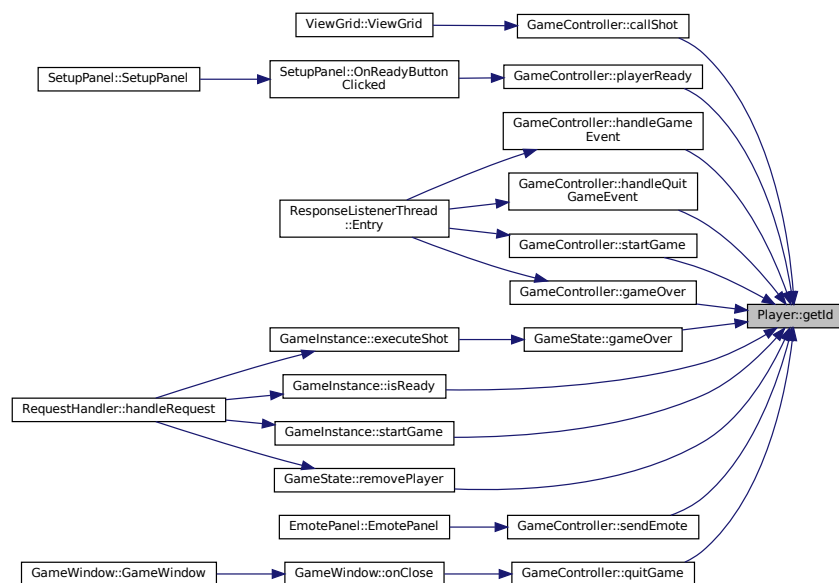
```
Player::Player (
    uuid playerId,
    std::string player_name )
```

4.29.2 Member Function Documentation

4.29.2.1 getId()

```
auto Player::getId ( ) const -> uuid
```

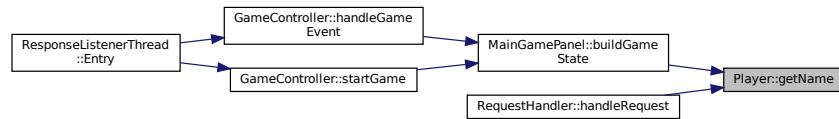
Here is the caller graph for this function:



4.29.2.2 getName()

```
auto Player::getName ( ) const -> std::string
```

Here is the caller graph for this function:



4.29.2.3 operator==()

```
bool Player::operator== (
    const Player & ) const [default]
```

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

- [/home/nico/Desktop/battleship/src/common/game_state/Player.h](#)
- [/home/nico/Desktop/battleship/src/common/game_state/Player.cpp](#)

4.30 PlayerGrid Class Reference

```
#include <PlayerGrid.h>
```

Public Member Functions

- [PlayerGrid](#) (uuid playerId, std::vector< [Ship](#) > shipsPlacement)

Public Attributes

- [uuid playerId](#)
id of owner of this grid
- std::vector< [Ship](#) > [shipsPlaced](#)
vector of all ships placed on this grid
- int [shotsReceived](#) [10][10] = {{0}}
shots received by this grid. 0 = tile not shot, 1 = miss, 2 = hit
- int [shotsFired](#) [10][10] = {{0}}
shots fired by this player. 0 = tile not shot, 1 = miss, 2 = hit

4.30.1 Detailed Description

Data container to store ocean grid info. A grid always belongs to one player (the one who has his ships on this grid)

4.30.2 Constructor & Destructor Documentation

4.30.2.1 PlayerGrid()

```
PlayerGrid::PlayerGrid (
    uuid playerId,
    std::vector< Ship > shipsPlacement )
```

4.30.3 Member Data Documentation

4.30.3.1 playerId

```
uuid PlayerGrid::playerId
```

id of owner of this grid

4.30.3.2 shipsPlaced

```
std::vector<Ship> PlayerGrid::shipsPlaced
```

vector of all ships placed on this grid

4.30.3.3 shotsFired

```
int PlayerGrid::shotsFired[10][10] = {{0}}
```

shots fired by this player. 0 = tile not shot, 1 = miss, 2 = hit

4.30.3.4 shotsReceived

```
int PlayerGrid::shotsReceived[10][10] = {{0}}
```

shots received by this grid. 0 = tile not shot, 1 = miss, 2 = hit

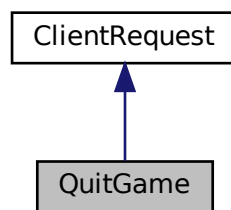
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.h](#)
- [/home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.cpp](#)

4.31 QuitGame Class Reference

```
#include <QuitGame.h>
```

Inheritance diagram for QuitGame:



Public Member Functions

- [QuitGame](#) ([uuid](#) playerId)

Additional Inherited Members

4.31.1 Constructor & Destructor Documentation

4.31.1.1 QuitGame()

```
QuitGame::QuitGame (
    uuid playerId ) [explicit]
```

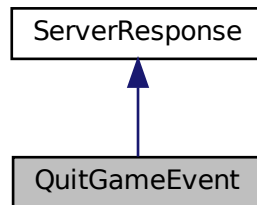
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/network/requests/QuitGame.h](#)
- [/home/nico/Desktop/battleship/src/common/network/requests/QuitGame.cpp](#)

4.32 QuitGameEvent Class Reference

```
#include <QuitGameEvent.h>
```

Inheritance diagram for QuitGameEvent:



Public Member Functions

- [QuitGameEvent](#) ([uuid](#) quitPlayerId)

Public Attributes

- const [uuid](#) quitPlayerId

Additional Inherited Members

4.32.1 Constructor & Destructor Documentation

4.32.1.1 QuitGameEvent()

```
QuitGameEvent::QuitGameEvent (
    uuid quitPlayerId ) [explicit]
```

4.32.2 Member Data Documentation

4.32.2.1 quitPlayerId

```
const uuid QuitGameEvent::quitPlayerId
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.h
- /home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.cpp

4.33 RequestHandler Class Reference

```
#include <RequestHandler.h>
```

Static Public Member Functions

- static std::unique_ptr< [ServerResponse](#) > [handleRequest](#) ([GameInstance](#) &gameInstance, const [ClientRequest](#) *const req)

4.33.1 Member Function Documentation

4.33.1.1 handleRequest()

```
std::unique_ptr< ServerResponse > RequestHandler::handleRequest (  
    GameInstance & gameInstance,  
    const ClientRequest *const req ) [static]
```

Handles an incoming request from the [ServerNetworkManager](#) and generates a response.

Parameters

<i>gameInstance</i>	
<i>req</i>	

Public Member Functions

- [ResponseListenerThread](#) (sockpp::tcp_connector *connection)

Protected Member Functions

- virtual ExitCode [Entry](#) ()

4.34.1 Detailed Description

Thread that is permanently listening for messages from the server

4.34.2 Constructor & Destructor Documentation

4.34.2.1 ResponseListenerThread()

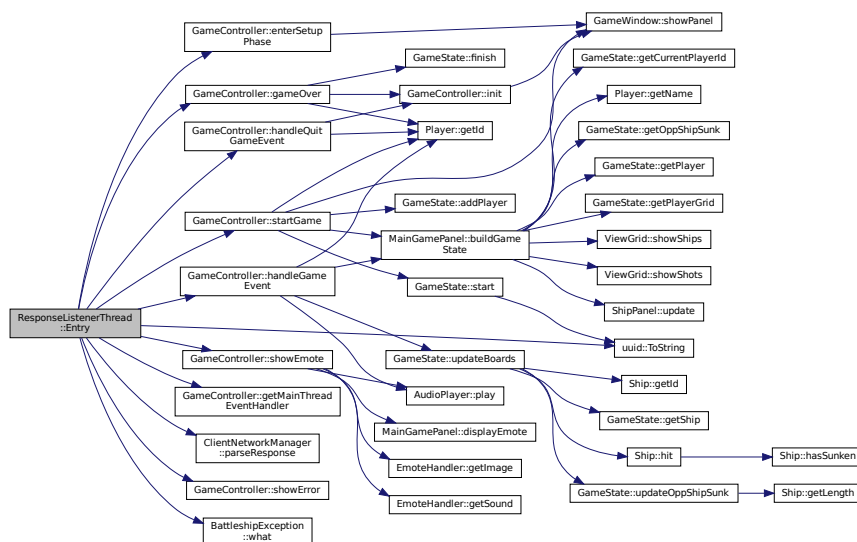
```
ResponseListenerThread::ResponseListenerThread (
    sockpp::tcp_connector * connection )
```

4.34.3 Member Function Documentation

4.34.3.1 Entry()

```
wxThread::ExitCode ResponseListenerThread::Entry () [protected], [virtual]
```

Here is the call graph for this function:



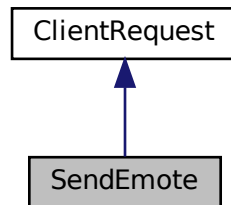
The documentation for this class was generated from the following files:

- /home/nico/Desktop/battleship/src/client/[ResponseListenerThread.h](#)
- /home/nico/Desktop/battleship/src/client/[ResponseListenerThread.cpp](#)

4.35 SendEmote Class Reference

```
#include <SendEmote.h>
```

Inheritance diagram for SendEmote:



Public Member Functions

- [SendEmote](#) ([uuid](#) playerId, [EmoteType](#) emote)
- auto [getEmote](#) () const -> [EmoteType](#)

Additional Inherited Members

4.35.1 Constructor & Destructor Documentation

4.35.1.1 SendEmote()

```
SendEmote::SendEmote (  
    uuid playerId,  
    EmoteType emote )
```

4.35.2 Member Function Documentation

4.35.2.1 getEmote()

```
auto SendEmote::getEmote ( ) const -> EmoteType
```

Here is the caller graph for this function:



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

- [/home/nico/Desktop/battleship/src/common/network/requests/SendEmote.h](#)
- [/home/nico/Desktop/battleship/src/common/network/requests/SendEmote.cpp](#)

4.36 ServerNetworkManager Class Reference

```
#include <ServerNetworkManager.h>
```

Public Member Functions

- [ServerNetworkManager](#) (uint16_t port)
- [~ServerNetworkManager](#) ()
- void [listenerLoop](#) ()

Static Public Member Functions

- static void [broadcastMessage](#) ([ServerResponse](#) &msg, const std::vector< [Player](#) > &players, const [Player](#) *exclude=nullptr)
- static void [onPlayerLeft](#) (uuid player_id)

4.36.1 Constructor & Destructor Documentation

4.36.1.1 ServerNetworkManager()

```
ServerNetworkManager::ServerNetworkManager (
    uint16_t port )
```

4.36.1.2 ~ServerNetworkManager()

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

4.36.2 Member Function Documentation

4.36.2.1 broadcastMessage()

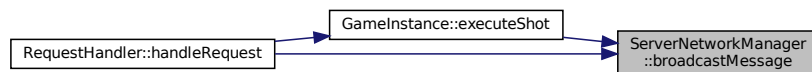
```
void ServerNetworkManager::broadcastMessage (
    ServerResponse & msg,
    const std::vector< Player > & players,
    const Player * exclude = nullptr ) [static]
```

Send out a [ServerResponse](#) to everyone

Parameters

<i>msg</i>	Response to be sent
<i>players</i>	Vector of all players/clients to send the message to
<i>exclude</i>	Optional player to exclude from the broadcast

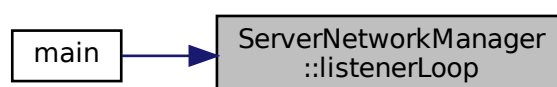
Here is the caller graph for this function:



4.36.2.2 listenerLoop()

```
void ServerNetworkManager::listenerLoop ( )
```

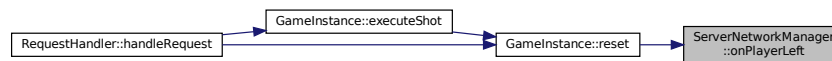
Here is the caller graph for this function:



4.36.2.3 onPlayerLeft()

```
void ServerNetworkManager::onPlayerLeft (
    uuid player_id ) [static]
```

Here is the caller graph for this function:



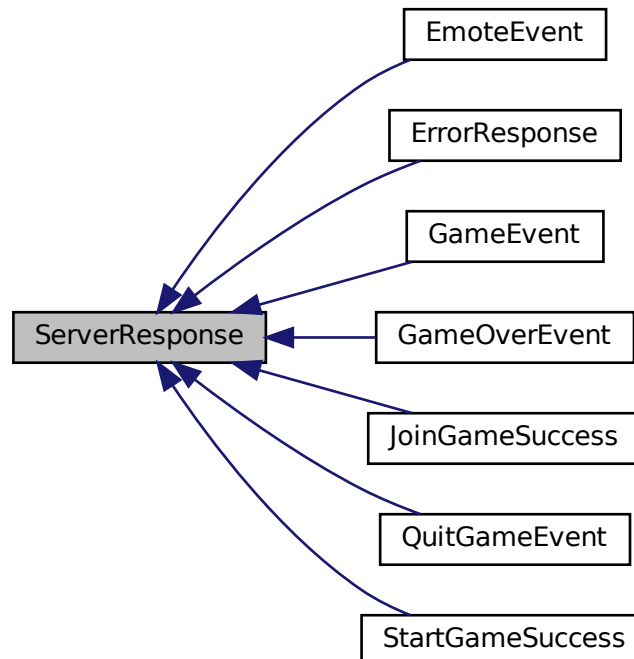
The documentation for this class was generated from the following files:

- /home/nico/Desktop/battleship/src/server/[ServerNetworkManager.h](#)
- /home/nico/Desktop/battleship/src/server/[ServerNetworkManager.cpp](#)

4.37 ServerResponse Class Reference

```
#include <ServerResponse.h>
```

Inheritance diagram for ServerResponse:



Public Member Functions

- auto `operator<=>` (const `ServerResponse` &) const =default
- virtual `~ServerResponse` ()=default

Public Attributes

- const `ResponseType responseType`

Protected Member Functions

- `ServerResponse` (`ResponseType responseType`)

4.37.1 Constructor & Destructor Documentation

4.37.1.1 `~ServerResponse()`

```
virtual ServerResponse::~~ServerResponse ( ) [virtual], [default]
```

4.37.1.2 `ServerResponse()`

```
ServerResponse::ServerResponse (
    ResponseType responseType ) [explicit], [protected]
```

4.37.2 Member Function Documentation

4.37.2.1 `operator<=>()`

```
auto ServerResponse::operator<=> (
    const ServerResponse & ) const [default]
```

4.37.3 Member Data Documentation

4.37.3.1 responseType

```
const ResponseType ServerResponse::responseType
```

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

- [/home/nico/Desktop/battleship/src/common/network/responses/ServerResponse.h](#)
- [/home/nico/Desktop/battleship/src/common/network/responses/ServerResponse.cpp](#)

4.38 SetupManager Class Reference

```
#include <SetupManager.h>
```

Public Member Functions

- [SetupManager](#) ()
constructor for [SetupManager](#). Initializes ships and grid.

Static Public Member Functions

- static bool [placeShip](#) (wxPoint &position, [Ship](#) *ship)
function places ship on grid and updates ship position
- static int * [getGrid](#) ()
getter for `_grid`
- static bool [placedAllShips](#) ()
checks if all ships have been placed (= no longer at initial position)

Static Public Attributes

- static std::vector< [Ship](#) > [_ships_placed](#)
- static [Ship](#) * [_selectedShip](#)

4.38.1 Constructor & Destructor Documentation

4.38.1.1 SetupManager()

```
SetupManager::SetupManager ( )
```

constructor for [SetupManager](#). Initializes ships and grid.

Here is the call graph for this function:



4.38.2 Member Function Documentation

4.38.2.1 getGrid()

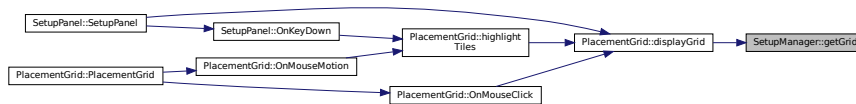
```
int * SetupManager::getGrid ( ) [static]
```

getter for _grid

Returns

int pointer to _grid array

Here is the caller graph for this function:



4.38.2.2 placedAllShips()

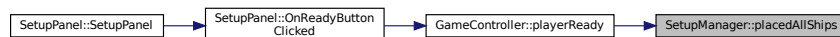
```
bool SetupManager::placedAllShips ( ) [static]
```

checks if all ships have been placed (= no longer at initial position)

Returns

true if all ships have been placed, false otherwise

Here is the caller graph for this function:



4.38.2.3 placeShip()

```
bool SetupManager::placeShip (
    wxPoint & position,
    Ship * ship ) [static]
```

function places ship on grid and updates ship position

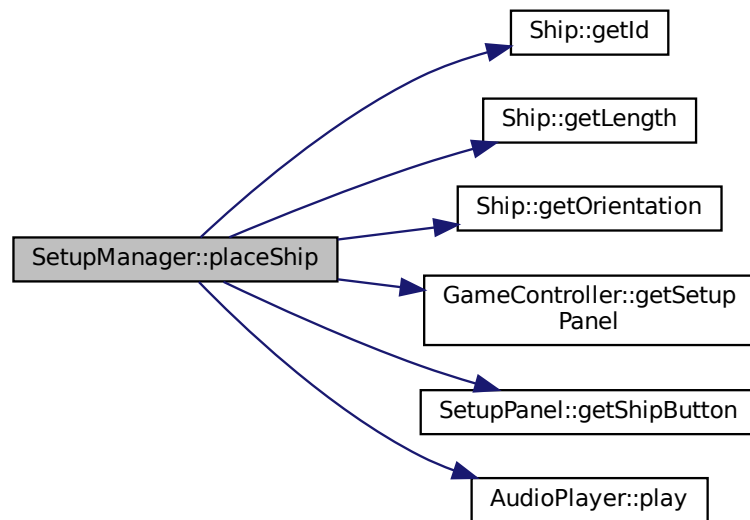
Parameters

<i>position</i>	of left-most or top-most cell of ship (depending on orientation)
<i>ship</i>	pointer to ship that should be placed

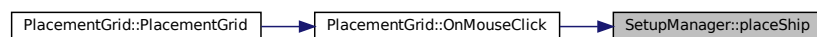
Returns

true if ship was placed successfully and false if ship couldn't be placed

Here is the call graph for this function:



Here is the caller graph for this function:



4.38.3 Member Data Documentation

4.38.3.1 `_selectedShip`

```
Ship * SetupManager::_selectedShip [static]
```

4.38.3.2 `_ships_placed`

```
std::vector< Ship > SetupManager::_ships_placed [static]
```

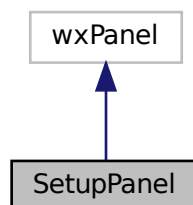
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/SetupManager.h](#)
- [/home/nico/Desktop/battleship/src/client/panels/SetupPanel.cpp](#)
- [/home/nico/Desktop/battleship/src/client/SetupManager.cpp](#)

4.39 SetupPanel Class Reference

```
#include <SetupPanel.h>
```

Inheritance diagram for SetupPanel:



Public Member Functions

- [SetupPanel](#) (wxWindow *parent)
Constructor of [SetupPanel](#). Creates the panel and all its components.
- void [OnReadyButtonClicked](#) (wxCommandEvent &event)
event handler for when the "Ready" button is clicked. Checks if all ships have been placed, and if so, notifies the [GameController](#) that the player is ready.
- void [OnKeyDown](#) (wxKeyEvent &event)
Key Event handler for rotating ship. Rotates selected ship if 'R' is pressed.
- wxStaticBitmap * [getShipButton](#) (int idx)
helper function used in [SetupManager::placeShip\(\)](#) to disable ship button after it has been placed
- wxButton * [getReadyButton](#) () const
getter for the ready button (for [GameController](#) to disable after it has been clicked)
- wxStaticText * [getReadyText](#) () const

4.39.1 Constructor & Destructor Documentation

4.39.1.1 SetupPanel()

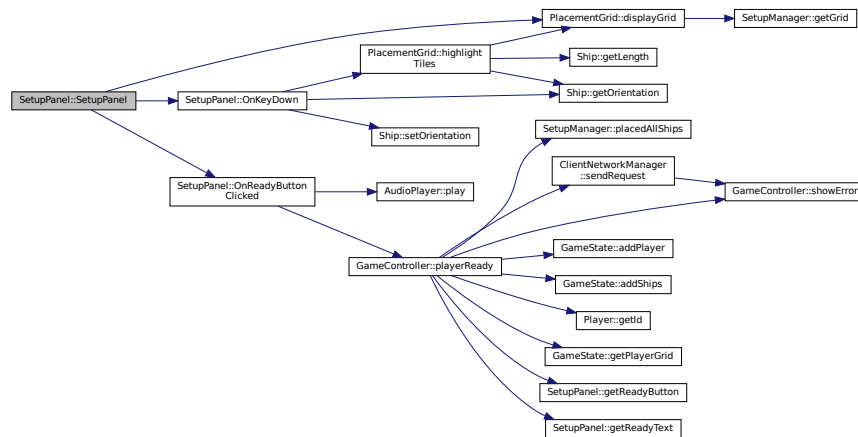
```
SetupPanel::SetupPanel (
    wxWindow * parent )
```

Constructor of [SetupPanel](#). Creates the panel and all its components.

Parameters

<i>parent</i>	
---------------	--

Here is the call graph for this function:



4.39.2 Member Function Documentation

4.39.2.1 getReadyButton()

```
wxButton * SetupPanel::getReadyButton ( ) const
```

getter for the ready button (for [GameController](#) to disable after it has been clicked)

Returns

wxButton* pointing to the ready button

Here is the caller graph for this function:



4.39.2.2 `getReadyText()`

```
wxStaticText * SetupPanel::getReadyText ( ) const
```

Here is the caller graph for this function:



4.39.2.3 `getShipButton()`

```
wxStaticBitmap * SetupPanel::getShipButton (
    int idx )
```

helper function used in [SetupManager::placeShip\(\)](#) to disable ship button after it has been placed

Parameters

<i>idx</i>	int in [0, 4] representing the index of the ship button to disable
------------	--

Returns

`wxStaticBitmap*` representing the ship button that should be disabled. Returns NULL if `idx` is invalid.

Here is the caller graph for this function:



4.39.2.4 `OnKeyDown()`

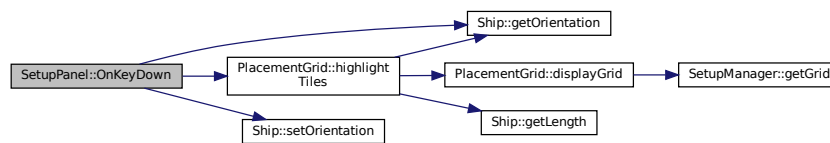
```
void SetupPanel::OnKeyDown (
    wxKeyEvent & event )
```

Key Event handler for rotating ship. Rotates selected ship if 'R' is pressed.

Parameters

<i>event</i>	
--------------	--

Here is the call graph for this function:



Here is the caller graph for this function:



4.39.2.5 OnReadyButtonClicked()

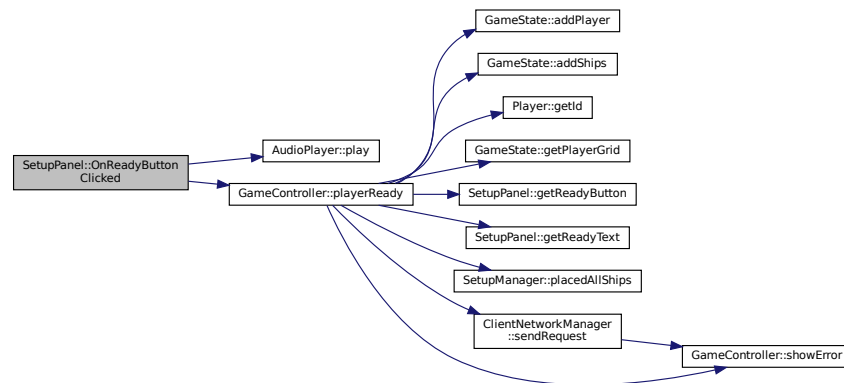
```
void SetupPanel::OnReadyButtonClicked (
    wxCommandEvent & event )
```

event handler for when the "Ready" button is clicked. Checks if all ships have been placed, and if so, notifies the [GameController](#) that the player is ready.

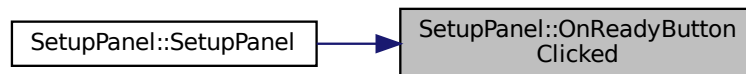
Parameters

<i>event</i>	<code>wxCommandEvent</code>
--------------	-----------------------------

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:

- </home/nico/Desktop/battleship/src/client/panels/SetupPanel.h>
- </home/nico/Desktop/battleship/src/client/panels/SetupPanel.cpp>

4.40 Ship Class Reference

```
#include <Ship.h>
```

Public Types

- enum class [Orientation](#) { [Vertical](#) , [Horizontal](#) }

Public Member Functions

- [Ship](#) (int length, [Coordinate](#) position, [Orientation](#) orientation, [uuid](#) id)
- auto [getLength](#) () const -> int
- auto [getPosition](#) () const -> [Coordinate](#)
- auto [getOrientation](#) () const -> [Orientation](#)
- auto [getId](#) () const -> [uuid](#)
- auto [setOrientation](#) ([Orientation](#) orientation) -> void
- auto [setPosition](#) ([Coordinate](#) position) -> void
- bool [hit](#) ([Coordinate](#) shot)
- bool [hasSunken](#) () const

4.40.1 Member Enumeration Documentation

4.40.1.1 Orientation

```
enum Ship::Orientation [strong]
```

Enumerator

Vertical	
Horizontal	

4.40.2 Constructor & Destructor Documentation

4.40.2.1 Ship()

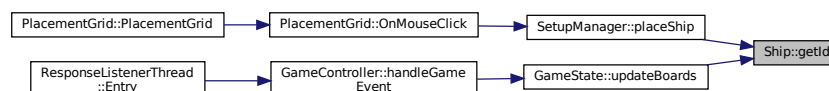
```
Ship::Ship (
    int length,
    Coordinate position,
    Orientation orientation,
    uuid id )
```

4.40.3 Member Function Documentation

4.40.3.1 getId()

```
auto Ship::getId ( ) const -> uuid
```

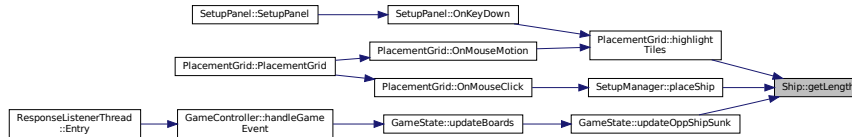
Here is the caller graph for this function:



4.40.3.2 `getLength()`

```
auto Ship::getLength ( ) const -> int
```

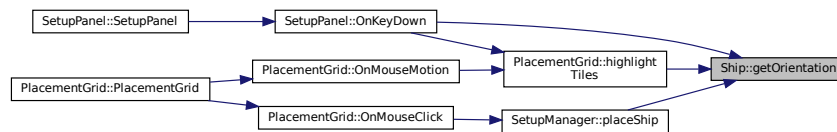
Here is the caller graph for this function:



4.40.3.3 `getOrientation()`

```
auto Ship::getOrientation ( ) const -> Orientation
```

Here is the caller graph for this function:



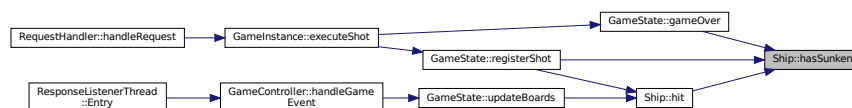
4.40.3.4 `getPosition()`

```
auto Ship::getPosition ( ) const -> Coordinate
```

4.40.3.5 `hasSunken()`

```
bool Ship::hasSunken ( ) const
```

Here is the caller graph for this function:



4.40.3.6 hit()

```
bool Ship::hit (
    Coordinate shot )
```

Hit detection. Processes a shot for an individual ship. Will update m_sunk and m_hits

Parameters

<i>shot</i>	coordinates of the shot called
-------------	--------------------------------

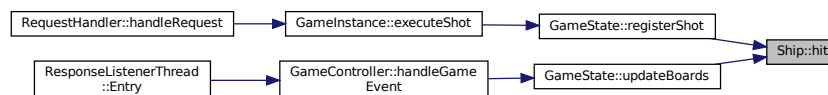
Returns

true if ship was hit, false otherwise

Here is the call graph for this function:



Here is the caller graph for this function:



4.40.3.7 setOrientation()

```
auto Ship::setOrientation (
    Ship::Orientation orientation ) -> void
```

Here is the caller graph for this function:



4.40.3.8 setPosition()

```
auto Ship::setPosition (
    Coordinate position ) -> void
```

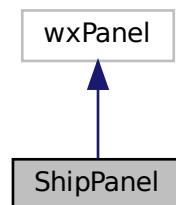
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/game_state/Ship.h](#)
- [/home/nico/Desktop/battleship/src/common/game_state/Ship.cpp](#)

4.41 ShipPanel Class Reference

```
#include <ShipPanel.h>
```

Inheritance diagram for ShipPanel:



Public Member Functions

- [ShipPanel](#) (wxWindow *parent, wxPoint pos, const std::array< bool, 5 > sunk)
- void [update](#) (const std::array< bool, 5 > sunk)

Updates the panel to show the ships that have been sunk.

4.41.1 Constructor & Destructor Documentation

4.41.1.1 ShipPanel()

```
ShipPanel::ShipPanel (
    wxWindow * parent,
    wxPoint pos,
    const std::array< bool, 5 > sunk )
```

4.41.2 Member Function Documentation

4.41.2.1 update()

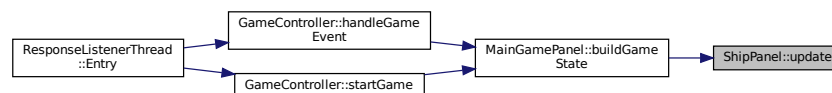
```
void ShipPanel::update (
    const std::array< bool, 5 > sunk )
```

Updates the panel to show the ships that have been sunk.

Parameters

<i>sunk</i>	An array of booleans, where true means that the ship has been sunk. The ships are sorted by length, the ship at index 0 is the longest one with length 5.
-------------	---

Here is the caller graph for this function:



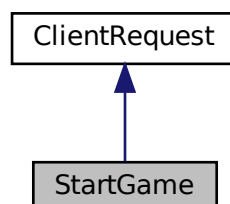
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.h](#)
- [/home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.cpp](#)

4.42 StartGame Class Reference

```
#include <StartGame.h>
```

Inheritance diagram for StartGame:



Public Member Functions

- [StartGame](#) ([uuid](#) playerId, std::vector< [Ship](#) > ships)
- auto [getShips](#) () const -> std::vector< [Ship](#) >

Additional Inherited Members

4.42.1 Constructor & Destructor Documentation

4.42.1.1 StartGame()

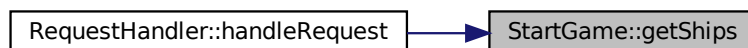
```
StartGame::StartGame (
    uuid playerId,
    std::vector< Ship > ships )
```

4.42.2 Member Function Documentation

4.42.2.1 getShips()

```
auto StartGame::getShips ( ) const -> std::vector<Ship>
```

Here is the caller graph for this function:



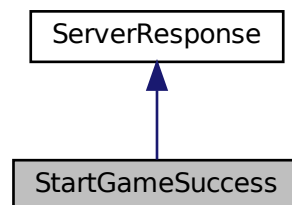
The documentation for this class was generated from the following files:

- `/home/nico/Desktop/battleship/src/common/network/requests/StartGame.h`
- `/home/nico/Desktop/battleship/src/common/network/requests/StartGame.cpp`

4.43 StartGameSuccess Class Reference

```
#include <StartGameSuccess.h>
```

Inheritance diagram for StartGameSuccess:



Public Member Functions

- `StartGameSuccess` (`std::vector< Player > players`, `uuid startingPlayerId`)

Public Attributes

- `const std::vector< Player > players`
- `const uuid startingPlayerId`

Additional Inherited Members

4.43.1 Constructor & Destructor Documentation

4.43.1.1 StartGameSuccess()

```
StartGameSuccess::StartGameSuccess (
    std::vector< Player > players,
    uuid startingPlayerId )
```

4.43.2 Member Data Documentation

4.43.2.1 players

```
const std::vector<Player> StartGameSuccess::players
```

4.43.2.2 startingPlayerId

```
const uuid StartGameSuccess::startingPlayerId
```

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

- /home/nico/Desktop/battleship/src/common/network/responses/[StartGameSuccess.h](#)
- /home/nico/Desktop/battleship/src/common/network/responses/[StartGameSuccess.cpp](#)

4.44 uuid Class Reference

```
#include <uuid.h>
```

Public Member Functions

- [uuid](#) ()=default
- [uuid](#) (const std::string &[uuid](#))
- auto [ToString](#) () const -> std::string
- bool [operator==](#) (const [uuid](#) &) const =default

Static Public Member Functions

- static auto [generateRandomUuid](#) () -> [uuid](#)

4.44.1 Constructor & Destructor Documentation

4.44.1.1 uuid() [1/2]

```
uuid::uuid ( ) [default]
```

4.44.1.2 uuid() [2/2]

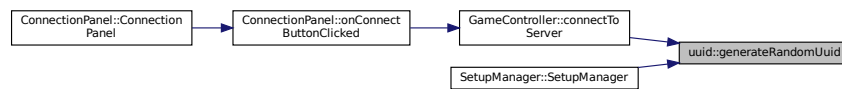
```
uuid::uuid (
    const std::string & uuid ) [explicit]
```


4.44.2 Member Function Documentation

4.44.2.1 generateRandomUuid()

```
auto uuid::generateRandomUuid ( ) -> uuid [static]
```

Here is the caller graph for this function:



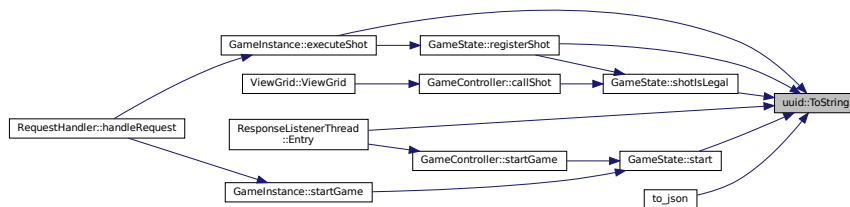
4.44.2.2 operator==()

```
bool uuid::operator== (
    const uuid & ) const [default]
```

4.44.2.3 ToString()

```
auto uuid::ToString ( ) const -> std::string
```

Here is the caller graph for this function:



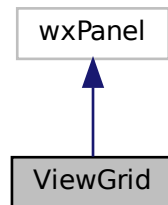
The documentation for this class was generated from the following files:

- [/home/nico/Desktop/battleship/src/common/uuid.h](#)
- [/home/nico/Desktop/battleship/src/common/uuid.cpp](#)

4.45 ViewGrid Class Reference

```
#include <ViewGrid.h>
```

Inheritance diagram for ViewGrid:



Public Types

- enum [GridType](#) { [own](#) , [opp](#) }

Public Member Functions

- [ViewGrid](#) (wxWindow *parent, [GridType](#) type)
- void [showShips](#) (const std::vector< [Ship](#) > &ships)
Displays the ships on the grid.
- void [showShots](#) (const int shots[10][10])
Displays the shots on the grid.

4.45.1 Member Enumeration Documentation

4.45.1.1 GridType

```
enum ViewGrid::GridType
```

Enumerator

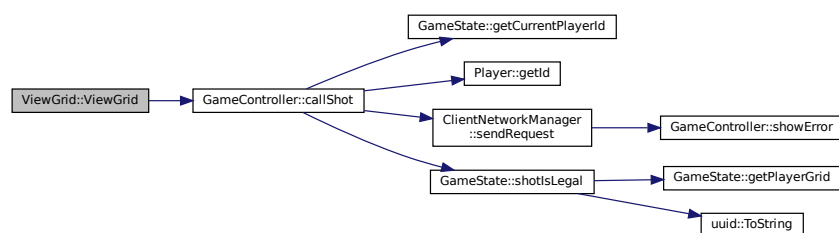
own	
opp	

4.45.2 Constructor & Destructor Documentation

4.45.2.1 ViewGrid()

```
ViewGrid::ViewGrid (
    wxWindow * parent,
    ViewGrid::GridType type )
```

Here is the call graph for this function:



4.45.3 Member Function Documentation

4.45.3.1 showShips()

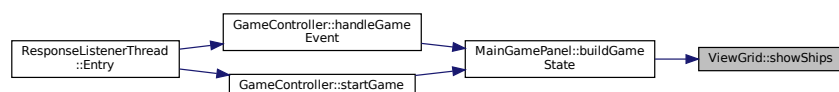
```
void ViewGrid::showShips (
    const std::vector< Ship > & ships )
```

Displays the ships on the grid.

Parameters

<i>ships</i>	A vector of ships that will be displayed. Displays the ships on the grid using the ship's position and orientation. This function should only be called for the own grid.
--------------	---

Here is the caller graph for this function:



4.45.3.2 showShots()

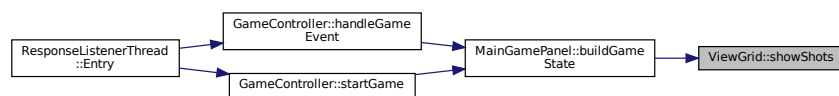
```
void ViewGrid::showShots (
    const int shots[10][10] )
```

Displays the shots on the grid.

Parameters

<i>shots</i>	A 2D array of integers, where 0 means no shot, 1 means a miss and 2 means a hit. A miss is marked with a white dot and a hit is marked with a red dot.
--------------	--

Here is the caller graph for this function:



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

- [/home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.h](#)
- [/home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.cpp](#)

Chapter 5

File Documentation

5.1 /home/nico/Desktop/battleship/src/client/AudioPlayer.cpp File Reference

```
#include "AudioPlayer.h"  
#include "Logger.h"
```

5.2 /home/nico/Desktop/battleship/src/client/AudioPlayer.h File Reference

```
#include "EmoteHandler.h"  
#include "wx/sound.h"  
#include <map>  
#include <string>  
#include <wx/wx.h>
```

Classes

- class [AudioPlayer](#)

5.3 /home/nico/Desktop/battleship/src/client/Battleship.cpp File Reference

```
#include "Battleship.h"  
#include "GameController.h"  
#include "GameWindow.h"  
#include "Logger.h"
```

5.4 /home/nico/Desktop/battleship/src/client/Battleship.h File Reference

```
#include <wx/wx.h>
```

Classes

- class [Battleship](#)

5.5 /home/nico/Desktop/battleship/src/client/ClientNetworkManager.cpp File Reference

```
#include "ClientNetworkManager.h"  
#include "GameController.h"  
#include "Logger.h"  
#include "ResponseListenerThread.h"  
#include "serialization/serialization.h"  
#include <nlohmann/json.hpp>  
#include <sockpp/exception.h>  
#include <sockpp/tcp_connector.h>  
#include <sstream>
```

5.6 /home/nico/Desktop/battleship/src/client/ClientNetworkManager.h File Reference

```
#include "ResponseListenerThread.h"  
#include "network/requests/ClientRequest.h"  
#include "network/responses/ServerResponse.h"  
#include "sockpp/tcp_connector.h"  
#include <memory>  
#include <string>
```

Classes

- class [ClientNetworkManager](#)

5.7 /home/nico/Desktop/battleship/src/client/EmoteHandler.cpp File Reference

```
#include "EmoteHandler.h"
```

5.8 /home/nico/Desktop/battleship/src/client/EmoteHandler.h File Reference

```
#include "network/responses/EmotionEvent.h"  
#include <map>  
#include <string>
```

Classes

- class [EmoteHandler](#)

5.9 /home/nico/Desktop/battleship/src/client/GameController.cpp File Reference

```
#include "GameController.h"  
#include "AudioPlayer.h"  
#include "ClientNetworkManager.h"  
#include "Logger.h"  
#include "network/requests/CallShot.h"  
#include "network/requests/JoinGame.h"  
#include "network/requests/QuitGame.h"  
#include "network/requests/SendEmote.h"  
#include "network/requests/StartGame.h"
```

5.10 /home/nico/Desktop/battleship/src/client/GameController.h File Reference

```
#include "EmoteHandler.h"  
#include "GameWindow.h"  
#include "SetupManager.h"  
#include "game_state/Player.h"  
#include "network/responses/EmotionEvent.h"  
#include "network/responses/GameEvent.h"  
#include "network/responses/QuitGameEvent.h"  
#include "network/responses/StartGameSuccess.h"  
#include "panels/ConnectionPanel.h"  
#include "panels/MainGamePanel.h"  
#include "panels/SetupPanel.h"  
#include <chrono>  
#include <wx/wx.h>
```

Classes

- class [GameController](#)

5.11 /home/nico/Desktop/battleship/src/client/GameWindow.cpp File Reference

```
#include "GameWindow.h"  
#include "../common/Logger.h"  
#include "GameController.h"  
#include <wx/wx.h>
```

5.12 /home/nico/Desktop/battleship/src/client/GameWindow.h File Reference

```
#include <wx/wx.h>
```

Classes

- class [GameWindow](#)

5.13 /home/nico/Desktop/battleship/src/client/main.cpp File Reference

```
#include "Battleship.h"  
#include <wx/wx.h>
```

Functions

- [wxIMPLEMENT_APP](#) ([Battleship](#))

5.13.1 Function Documentation

5.13.1.1 wxIMPLEMENT_APP()

```
wxIMPLEMENT_APP (  
    Battleship )
```

5.14 /home/nico/Desktop/battleship/src/server/main.cpp File Reference

```
#include "Logger.h"  
#include "ServerNetworkManager.h"
```


Functions

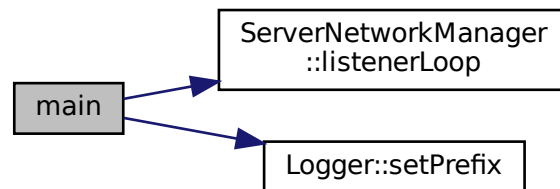
- int [main](#) ()

5.14.1 Function Documentation

5.14.1.1 main()

```
int main ( )
```

Here is the call graph for this function:



5.15 /home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.cpp File Reference

```
#include "ConnectionPanel.h"
#include "../AudioPlayer.h"
#include "../GameController.h"
#include "../uiElements/ImagePanel.h"
```

5.16 /home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.h File Reference

```
#include "../uiElements/InputField.h"
#include <wx/wx.h>
```

Classes

- class [ConnectionPanel](#)

5.17 /home/nico/Desktop/battleship/src/client/panels/MainGamePanel.cpp File Reference

```
#include "MainGamePanel.h"
#include "../GameController.h"
#include "Logger.h"
```

5.18 /home/nico/Desktop/battleship/src/client/panels/MainGamePanel.h File Reference

```
#include "../../common/game_state/Player.h"
#include "../../common/network/responses/EmoteEvent.h"
#include "../uiElements/EmotePanel.h"
#include "../uiElements/EmotePopup.h"
#include "../uiElements/ShipPanel.h"
#include "../uiElements/ViewGrid.h"
#include "game_state/GameState.h"
#include <wx/wx.h>
```

Classes

- class [MainGamePanel](#)

5.19 /home/nico/Desktop/battleship/src/client/panels/SetupPanel.cpp File Reference

```
#include "SetupPanel.h"
#include "../AudioPlayer.h"
#include "../GameController.h"
#include "Logger.h"
```

5.20 /home/nico/Desktop/battleship/src/client/panels/SetupPanel.h File Reference

```
#include "../SetupManager.h"
#include "../uiElements/PlacementGrid.h"
#include <wx/wx.h>
```

Classes

- class [SetupPanel](#)

5.21 /home/nico/Desktop/battleship/src/client/ResponseListenerThread.cpp File Reference

```
#include "ResponseListenerThread.h"
#include "ClientNetworkManager.h"
#include "GameController.h"
#include "Logger.h"
#include "network/responses/ErrorResponse.h"
#include "network/responses/GameOverEvent.h"
#include <sstream>
```

5.22 /home/nico/Desktop/battleship/src/client/ResponseListenerThread.h File Reference

```
#include <sockpp/tcp_connector.h>
#include <wx/wx.h>
```

Classes

- class [ResponseListenerThread](#)

5.23 /home/nico/Desktop/battleship/src/client/SetupManager.cpp File Reference

```
#include "SetupManager.h"
#include "AudioPlayer.h"
#include "GameController.h"
#include "Logger.h"
#include "game_state/Coordinate.h"
```

5.24 /home/nico/Desktop/battleship/src/client/SetupManager.h File Reference

```
#include "../common/game_state/Ship.h"
#include <vector>
#include <wx/gdicmn.h>
```

Classes

- class [SetupManager](#)

5.25 /home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.cpp File Reference

```
#include "EmotePanel.h"  
#include "../GameController.h"  
#include "Logger.h"
```

5.26 /home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.h File Reference

```
#include "../../../common/network/responses/EmotionEvent.h"  
#include "../EmoteHandler.h"  
#include <chrono>  
#include <wx/wx.h>
```

Classes

- class [EmotePanel](#)

5.27 /home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.cpp File Reference

```
#include "EmotePopup.h"  
#include "Logger.h"
```

5.28 /home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.h File Reference

```
#include "../EmoteHandler.h"  
#include <wx/popupwin.h>  
#include <wx/wx.h>
```

Classes

- class [EmotePopup](#)

5.29 /home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.cpp File Reference

```
#include "ImagePanel.h"
```

5.30 /home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.h File Reference

```
#include <wx/sizer.h>
#include <wx/wx.h>
```

Classes

- class [ImagePanel](#)

5.31 /home/nico/Desktop/battleship/src/client/uiElements/InputField.cpp File Reference

```
#include "InputField.h"
```

5.32 /home/nico/Desktop/battleship/src/client/uiElements/InputField.h File Reference

```
#include <wx/wx.h>
```

Classes

- class [InputField](#)

5.33 /home/nico/Desktop/battleship/src/client/uiElements/Placement↵ Grid.cpp File Reference

```
#include "PlacementGrid.h"
#include "../SetupManager.h"
#include "Logger.h"
#include <wx/wx.h>
```

5.34 /home/nico/Desktop/battleship/src/client/uiElements/Placement↵ Grid.h File Reference

```
#include <wx/wx.h>
```

Classes

- class [PlacementGrid](#)

5.35 /home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.cpp File Reference

```
#include "ShipPanel.h"
```

5.36 /home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.h File Reference

```
#include "game_state/Ship.h"  
#include <string>  
#include <wx/wx.h>
```

Classes

- class [ShipPanel](#)

5.37 /home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.cpp File Reference

```
#include "ViewGrid.h"  
#include "../GameController.h"  
#include "Logger.h"
```

5.38 /home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.h File Reference

```
#include "../../../common/game_state/Ship.h"  
#include <wx/wx.h>
```

Classes

- class [ViewGrid](#)

5.39 /home/nico/Desktop/battleship/src/common/exceptions/BattleshipException.h File Reference

```
#include <string>
```

Classes

- class [BattleshipException](#)

5.40 /home/nico/Desktop/battleship/src/common/game_state/Coordinate.h File Reference

```
#include <compare>
```

Classes

- struct [Coordinate](#)

5.41 /home/nico/Desktop/battleship/src/common/game_state/GameState.cpp File Reference

```
#include "GameState.h"  
#include "Coordinate.h"  
#include "Logger.h"  
#include "Player.h"  
#include <cassert>  
#include <stdexcept>  
#include <utility>
```

5.42 /home/nico/Desktop/battleship/src/common/game_state/GameState.h File Reference

```
#include "Coordinate.h"  
#include "game_state/Player.h"  
#include "game_state/PlayerGrid.h"  
#include "network/responses/GameEvent.h"  
#include <vector>
```

Classes

- class [GameState](#)

5.43 /home/nico/Desktop/battleship/src/common/game_state/Player.cpp File Reference

```
#include "Player.h"  
#include <utility>
```

5.44 /home/nico/Desktop/battleship/src/common/game_state/Player.h File Reference

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

Classes

- class [Player](#)

5.45 /home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.cpp File Reference ↩

```
#include "game_state/PlayerGrid.h"  
#include "game_state/Ship.h"  
#include "uuid.h"  
#include <utility>  
#include <vector>
```

5.46 /home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.h File Reference ↩

```
#include "game_state/Ship.h"  
#include "uuid.h"  
#include <vector>
```

Classes

- class [PlayerGrid](#)

5.47 /home/nico/Desktop/battleship/src/common/game_state/Ship.cpp File Reference

```
#include "Ship.h"
```


5.48 /home/nico/Desktop/battleship/src/common/game_state/Ship.h File Reference

```
#include "../uuid.h"  
#include "Coordinate.h"  
#include <vector>
```

Classes

- class [Ship](#)

5.49 /home/nico/Desktop/battleship/src/common/Logger.cpp File Reference

```
#include "Logger.h"  
#include <sstream>
```

5.50 /home/nico/Desktop/battleship/src/common/Logger.h File Reference

```
#include "exceptions/BattleshipException.h"  
#include <ctime>  
#include <fstream>  
#include <iomanip>  
#include <iostream>  
#include <string>
```

Classes

- class [Logger](#)

Macros

- #define [LOG](#)(message) [Logger::log](#)(message, __func__)

5.50.1 Macro Definition Documentation

5.50.1.1 LOG

```
#define LOG(  
    message ) Logger::log(message, __func__)
```

5.51 [/home/nico/Desktop/battleship/src/common/network/requests/CallShot.cpp](#) File Reference

```
#include "CallShot.h"  
#include "../game_state/Coordinate.h"  
#include "../uuid.h"  
#include "ClientRequest.h"
```

5.52 [/home/nico/Desktop/battleship/src/common/network/requests/CallShot.h](#) File Reference

```
#include "ClientRequest.h"  
#include "game_state/Coordinate.h"
```

Classes

- class [CallShot](#)

5.53 [/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.cpp](#) File Reference

```
#include "ClientRequest.h"  
#include "uuid.h"
```

5.54 [/home/nico/Desktop/battleship/src/common/network/requests/ClientRequest.h](#) File Reference

```
#include "uuid.h"  
#include <compare>
```

Classes

- class [ClientRequest](#)

Enumerations

- enum class [RequestType](#) {
 [JoinGame](#) , [StartGame](#) , [CallShot](#) , [SendEmote](#) ,
 [QuitGame](#) , [PlayAgain](#) }

5.54.1 Enumeration Type Documentation

5.54.1.1 RequestType

```
enum RequestType [strong]
```

Enumerator

JoinGame	
StartGame	
CallShot	
SendEmote	
QuitGame	
PlayAgain	

5.55 /home/nico/Desktop/battleship/src/common/network/requests/↵ JoinGame.cpp File Reference

```
#include "JoinGame.h"  
#include "../..//uuid.h"  
#include "ClientRequest.h"  
#include <string>  
#include <utility>
```

5.56 /home/nico/Desktop/battleship/src/common/network/requests/↵ JoinGame.h File Reference

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

Classes

- class [JoinGame](#)

5.57 [/home/nico/Desktop/battleship/src/common/network/requests/](#) PlayAgain.cpp File Reference

```
#include "PlayAgain.h"  
#include "../..//uuid.h"  
#include "ClientRequest.h"
```

5.58 [/home/nico/Desktop/battleship/src/common/network/requests/](#) PlayAgain.h File Reference

```
#include "ClientRequest.h"
```

Classes

- class [PlayAgain](#)

5.59 [/home/nico/Desktop/battleship/src/common/network/requests/](#) QuitGame.cpp File Reference

```
#include "QuitGame.h"  
#include "../..//uuid.h"  
#include "ClientRequest.h"
```

5.60 [/home/nico/Desktop/battleship/src/common/network/requests/](#) QuitGame.h File Reference

```
#include "ClientRequest.h"
```

Classes

- class [QuitGame](#)

5.61 [/home/nico/Desktop/battleship/src/common/network/requests/](#) SendEmote.cpp File Reference

```
#include "SendEmote.h"  
#include "../..//uuid.h"  
#include "ClientRequest.h"  
#include <string>  
#include <utility>
```

5.62 /home/nico/Desktop/battleship/src/common/network/requests/↵ SendEmote.h File Reference

```
#include "ClientRequest.h"  
#include "network/responses/EmoteEvent.h"  
#include <string>
```

Classes

- class [SendEmote](#)

5.63 /home/nico/Desktop/battleship/src/common/network/requests/↵ StartGame.cpp File Reference

```
#include "StartGame.h"  
#include "../game_state/Ship.h"  
#include "../uuid.h"  
#include "ClientRequest.h"  
#include <utility>  
#include <vector>
```

5.64 /home/nico/Desktop/battleship/src/common/network/requests/↵ StartGame.h File Reference

```
#include "../game_state/Ship.h"  
#include "ClientRequest.h"  
#include <vector>
```

Classes

- class [StartGame](#)

5.65 /home/nico/Desktop/battleship/src/common/network/responses/↵ EmoteEvent.cpp File Reference

```
#include "EmoteEvent.h"
```

5.66 /home/nico/Desktop/battleship/src/common/network/responses/↵ EmotionEvent.h File Reference

```
#include "game_state/Player.h"  
#include "network/responses/ServerResponse.h"  
#include <array>
```

Classes

- class [EmotionEvent](#)

Enumerations

- enum class [EmoteType](#) {
 [MiddleFinger](#) , [RussianWarshipGoFuckYourself](#) , [Mocking](#) , [BestPirate](#) ,
 [Panic](#) , [Clown](#) }

5.66.1 Enumeration Type Documentation

5.66.1.1 EmoteType

```
enum EmoteType [strong]
```

Enumerator

MiddleFinger	
RussianWarshipGoFuckYourself	
Mocking	
BestPirate	
Panic	
Clown	

5.67 /home/nico/Desktop/battleship/src/common/network/responses/↵ ErrorResponse.cpp File Reference

```
#include "ErrorResponse.h"  
#include <utility>
```

5.68 /home/nico/Desktop/battleship/src/common/network/responses/↵ ErrorResponse.h File Reference

```
#include "exceptions/BattleshipException.h"  
#include "network/responses/ServerResponse.h"
```

Classes

- class [ErrorResponse](#)

5.69 /home/nico/Desktop/battleship/src/common/network/responses/↵ GameEvent.cpp File Reference

```
#include "GameEvent.h"  
#include "game_state/Coordinate.h"  
#include "game_state/Ship.h"  
#include "uuid.h"
```

5.70 /home/nico/Desktop/battleship/src/common/network/responses/↵ GameEvent.h File Reference

```
#include "game_state/Coordinate.h"  
#include "game_state/Ship.h"  
#include "network/responses/ServerResponse.h"  
#include "uuid.h"
```

Classes

- class [GameEvent](#)

5.71 /home/nico/Desktop/battleship/src/common/network/responses/↵ GameOverEvent.cpp File Reference

```
#include "GameOverEvent.h"  
#include "uuid.h"
```

5.72 [/home/nico/Desktop/battleship/src/common/network/responses/GameOverEvent.h](#) File Reference

```
#include "network/responses/ServerResponse.h"
#include "uuid.h"
```

Classes

- class [GameOverEvent](#)

5.73 [/home/nico/Desktop/battleship/src/common/network/responses/JoinGameSuccess.cpp](#) File Reference

```
#include "JoinGameSuccess.h"
```

5.74 [/home/nico/Desktop/battleship/src/common/network/responses/JoinGameSuccess.h](#) File Reference

```
#include "network/requests/ClientRequest.h"
#include "network/responses/ServerResponse.h"
#include "uuid.h"
```

Classes

- class [JoinGameSuccess](#)

5.75 [/home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.cpp](#) File Reference

```
#include "QuitGameEvent.h"
#include "ServerResponse.h"
```

5.76 [/home/nico/Desktop/battleship/src/common/network/responses/QuitGameEvent.h](#) File Reference

```
#include "ServerResponse.h"
#include "uuid.h"
```


Classes

- class [QuitGameEvent](#)

5.77 /home/nico/Desktop/battleship/src/common/network/responses/↵ ServerResponse.cpp File Reference

```
#include "ServerResponse.h"
```

5.78 /home/nico/Desktop/battleship/src/common/network/responses/↵ ServerResponse.h File Reference

```
#include <compare>
```

Classes

- class [ServerResponse](#)

Enumerations

- enum class [ResponseType](#) {
[GameEvent](#) , [EmoteEvent](#) , [JoinGameSuccess](#) , [StartGameSuccess](#) ,
[GameOverEvent](#) , [ErrorResponse](#) , [QuitGameEvent](#) }

5.78.1 Enumeration Type Documentation

5.78.1.1 ResponseType

```
enum ResponseType [strong]
```

Enumerator

GameEvent	
EmoteEvent	
JoinGameSuccess	
StartGameSuccess	
GameOverEvent	
ErrorResponse	
QuitGameEvent	

5.79 [/home/nico/Desktop/battleship/src/common/network/responses/](#) StartGameSuccess.cpp File Reference

```
#include "StartGameSuccess.h"
#include <utility>
```

5.80 [/home/nico/Desktop/battleship/src/common/network/responses/](#) StartGameSuccess.h File Reference

```
#include "game_state/Player.h"
#include "network/responses/ServerResponse.h"
#include <vector>
```

Classes

- class [StartGameSuccess](#)

5.81 [/home/nico/](#) [Desktop/battleship/src/common/serialization/serialization.h](#) File Reference

```
#include "game_state/Coordinate.h"
#include "game_state/Ship.h"
#include "network/requests/CallShot.h"
#include "network/requests/ClientRequest.h"
#include "network/requests/JoinGame.h"
#include "network/requests/PlayAgain.h"
#include "network/requests/QuitGame.h"
#include "network/requests/SendEmote.h"
#include "network/requests/StartGame.h"
#include "network/responses/EmoteEvent.h"
#include "network/responses/ErrorResponse.h"
#include "network/responses/GameEvent.h"
#include "network/responses/GameOverEvent.h"
#include "network/responses/JoinGameSuccess.h"
#include "network/responses/QuitGameEvent.h"
#include "network/responses/ServerResponse.h"
#include "network/responses/StartGameSuccess.h"
#include <memory>
#include <nlohmann/json.hpp>
```

Functions

- void [to_json](#) (nlohmann::json &json, const [Coordinate](#) &position)
- void [from_json](#) (const nlohmann::json &json, [Coordinate](#) &position)
- void [to_json](#) (nlohmann::json &json, const [uuid](#) &uuid)
- void [from_json](#) (const nlohmann::json &json, [uuid](#) &uuid_v)
- [NLOHMANN_JSON_SERIALIZE_ENUM](#) ([Ship::Orientation](#), {{[Ship::Orientation::Vertical](#), "v"}, {[Ship::Orientation::Horizontal](#), "h"}}) [NLOHMANN_JSON_SERIALIZE_ENUM](#)([RequestType](#)

5.81.1 Function Documentation

5.81.1.1 from_json() [1/2]

```
void from_json (
    const nlohmann::json & json,
    Coordinate & position ) [inline]
```

5.81.1.2 from_json() [2/2]

```
void from_json (
    const nlohmann::json & json,
    uuid & uuid_v ) [inline]
```

5.81.1.3 NLOHMANN_JSON_SERIALIZE_ENUM()

```
NLOHMANN_JSON_SERIALIZE_ENUM (
    Ship::Orientation ,
    {{Ship::Orientation::Vertical, "v"}, {Ship::Orientation::Horizontal, "h"}} )
```

5.81.1.4 to_json() [1/2]

```
void to_json (
    nlohmann::json & json,
    const Coordinate & position ) [inline]
```

5.81.1.5 to_json() [2/2]

```
void to_json (
    nlohmann::json & json,
    const uuid & uuid ) [inline]
```

Here is the call graph for this function:



5.82 /home/nico/Desktop/battleship/src/common/uuid.cpp File Reference

```
#include "uuid.h"
#include <algorithm>
#include <cassert>
#include <iomanip>
#include <ios>
#include <random>
#include <sstream>
#include <string>
```

5.83 /home/nico/Desktop/battleship/src/common/uuid.h File Reference

```
#include <array>
#include <cstdint>
#include <string>
```

Classes

- class [uuid](#)
- struct [std::hash< uuid >](#)

5.84 /home/nico/Desktop/battleship/src/server/GameInstance.cpp File Reference

```
#include "GameInstance.h"
#include "Logger.h"
#include "ServerNetworkManager.h"
#include "network/responses/ErrorResponse.h"
#include "network/responses/GameEvent.h"
#include "network/responses/GameOverEvent.h"
#include "network/responses/JoinGameSuccess.h"
#include "network/responses/QuitGameEvent.h"
#include "network/responses/ServerResponse.h"
#include <cassert>
```

5.85 /home/nico/Desktop/battleship/src/server/GameInstance.h File Reference

```
#include "game_state/GameState.h"
#include "game_state/Player.h"
#include "network/requests/CallShot.h"
```

```
#include "network/requests/JoinGame.h"
#include "network/requests/QuitGame.h"
#include <mutex>
#include <string>
#include <unordered_map>
#include <vector>
```

Classes

- class [GameInstance](#)

5.86 /home/nico/Desktop/battleship/src/server/RequestHandler.cpp File Reference

```
#include "RequestHandler.h"
#include "GameInstance.h"
#include "Logger.h"
#include "ServerNetworkManager.h"
#include "network/requests/CallShot.h"
#include "network/requests/JoinGame.h"
#include "network/requests/QuitGame.h"
#include "network/requests/SendEmote.h"
#include "network/requests/StartGame.h"
#include "network/responses/ErrorResponse.h"
#include "network/responses/JoinGameSuccess.h"
#include "network/responses/QuitGameEvent.h"
#include "network/responses/ServerResponse.h"
#include "network/responses/StartGameSuccess.h"
```

5.87 /home/nico/Desktop/battleship/src/server/RequestHandler.h File Reference

```
#include "GameInstance.h"
#include "network/requests/ClientRequest.h"
#include "network/responses/ServerResponse.h"
#include <memory>
```

Classes

- class [RequestHandler](#)

5.88 /home/nico/Desktop/battleship/src/server/ServerNetworkManager.cpp File Reference

```
#include "ServerNetworkManager.h"
#include "Logger.h"
#include "RequestHandler.h"
#include "network/responses/ServerResponse.h"
#include "serialization/serialization.h"
#include <nlohmann/json.hpp>
#include <sstream>
#include <string>
```

5.89 /home/nico/Desktop/battleship/src/server/ServerNetworkManager.h File Reference

```
#include "GameInstance.h"
#include "game_state/Player.h"
#include "network/responses/ServerResponse.h"
#include "sockpp/tcp_acceptor.h"
#include "sockpp/tcp_connector.h"
#include "sockpp/tcp_socket.h"
#include <functional>
#include <shared_mutex>
#include <thread>
#include <unordered_map>
```

Classes

- class [ServerNetworkManager](#)

Index

/home/nico/Desktop/battleship/src/client/AudioPlayer.cpp, 111 /home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.cpp, 118
/home/nico/Desktop/battleship/src/client/AudioPlayer.h, 111 /home/nico/Desktop/battleship/src/client/uiElements/EmotePopup.h, 118
/home/nico/Desktop/battleship/src/client/Battleship.cpp, 111 /home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.cpp, 118
/home/nico/Desktop/battleship/src/client/Battleship.h, 112 /home/nico/Desktop/battleship/src/client/uiElements/ImagePanel.h, 119
/home/nico/Desktop/battleship/src/client/ClientNetworkManager.cpp, 112 /home/nico/Desktop/battleship/src/client/uiElements/TextField.cpp, 119
/home/nico/Desktop/battleship/src/client/ClientNetworkManager.h, 112 /home/nico/Desktop/battleship/src/client/uiElements/TextField.h, 119
/home/nico/Desktop/battleship/src/client/EmoteHandler.cpp, 112 /home/nico/Desktop/battleship/src/client/uiElements/PlacementGrid.cpp, 119
/home/nico/Desktop/battleship/src/client/EmoteHandler.h, 113 /home/nico/Desktop/battleship/src/client/uiElements/PlacementGrid.h, 119
/home/nico/Desktop/battleship/src/client/GameController.cpp, 113 /home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.cpp, 120
/home/nico/Desktop/battleship/src/client/GameController.h, 113 /home/nico/Desktop/battleship/src/client/uiElements/ShipPanel.h, 120
/home/nico/Desktop/battleship/src/client/GameWindow.cpp, 114 /home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.cpp, 120
/home/nico/Desktop/battleship/src/client/GameWindow.h, 114 /home/nico/Desktop/battleship/src/client/uiElements/ViewGrid.h, 120
/home/nico/Desktop/battleship/src/client/ResponseListenerThread.cpp, 117 /home/nico/Desktop/battleship/src/common/Logger.cpp, 123
/home/nico/Desktop/battleship/src/client/ResponseListenerThread.h, 117 /home/nico/Desktop/battleship/src/common/Logger.h, 123
/home/nico/Desktop/battleship/src/client/SetupManager.cpp, 117 /home/nico/Desktop/battleship/src/common/exceptions/BattleshipException.h, 121
/home/nico/Desktop/battleship/src/client/SetupManager.h, 117 /home/nico/Desktop/battleship/src/common/game_state/Coordinate.h, 121
/home/nico/Desktop/battleship/src/client/main.cpp, 114 /home/nico/Desktop/battleship/src/common/game_state/GameState.cpp, 121
/home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.cpp, 115 /home/nico/Desktop/battleship/src/common/game_state/GameState.h, 121
/home/nico/Desktop/battleship/src/client/panels/ConnectionPanel.h, 115 /home/nico/Desktop/battleship/src/common/game_state/Player.cpp, 122
/home/nico/Desktop/battleship/src/client/panels/MainGamePanel.cpp, 116 /home/nico/Desktop/battleship/src/common/game_state/Player.h, 122
/home/nico/Desktop/battleship/src/client/panels/MainGamePanel.h, 116 /home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.cpp, 122
/home/nico/Desktop/battleship/src/client/panels/SetupPanel.cpp, 116 /home/nico/Desktop/battleship/src/common/game_state/PlayerGrid.h, 122
/home/nico/Desktop/battleship/src/client/panels/SetupPanel.h, 116 /home/nico/Desktop/battleship/src/common/game_state/Ship.cpp, 122
/home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.cpp, 118 /home/nico/Desktop/battleship/src/common/game_state/Ship.h, 123
/home/nico/Desktop/battleship/src/client/uiElements/EmotePanel.h, 118 /home/nico/Desktop/battleship/src/common/network/requests/CallShot.cpp, 123

- CallShot, 11
 - CallShot, 12
 - ClientRequest.h, 125
 - getPosition, 12
- callShot
 - GameController, 27
- Cannon
 - AudioPlayer, 7
- cellX_prev
 - PlacementGrid, 76
- cellY_prev
 - PlacementGrid, 76
- ClientNetworkManager, 12
 - init, 12
 - parseResponse, 13
 - sendRequest, 13
- ClientRequest, 14
 - ~ClientRequest, 15
 - ClientRequest, 15
 - getPlayerId, 15
 - getRequestType, 15
- ClientRequest.h
 - CallShot, 125
 - JoinGame, 125
 - PlayAgain, 125
 - QuitGame, 125
 - RequestType, 125
 - SendEmote, 125
 - StartGame, 125
- ClientState
 - GameState, 48
- Clip
 - AudioPlayer, 7
- Clown
 - EmoteEvent.h, 128
- ConnectionPanel, 16
 - ConnectionPanel, 17
 - getServerAddress, 17
 - getServerPort, 18
 - getUserName, 18
 - onConnectButtonClicked, 19
- connectToServer
 - GameController, 28
- Coordinate, 20
 - operator<=>, 20
 - x, 20
 - y, 21
- displayEmote
 - MainGamePanel, 71
- displayGrid
 - PlacementGrid, 73
- emote
 - EmoteEvent, 22
- EmoteEvent, 21
 - emote, 22
 - EmoteEvent, 21
 - playerId, 22
- ServerResponse.h, 131
- EmoteEvent.h
 - BestPirate, 128
 - Clown, 128
 - EmoteType, 128
 - MiddleFinger, 128
 - Mocking, 128
 - Panic, 128
 - RussianWarshipGoFuckYourself, 128
- EmoteHandler, 22
 - getImage, 22
 - getImageLarge, 23
 - getSound, 23
- EmotePanel, 24
 - EmotePanel, 24
- EmotePopup, 25
 - EmotePopup, 25
- EmoteType
 - EmoteEvent.h, 128
- enterSetupPhase
 - GameController, 29
- Entry
 - ResponseListenerThread, 85
- ErrorResponse, 26
 - ErrorResponse, 26
 - exception, 26
 - ServerResponse.h, 131
- exception
 - ErrorResponse, 26
- executeShot
 - GameInstance, 40
- finish
 - GameState, 49
- Finished
 - GameState, 47
- from_json
 - serialization.h, 133
- GameController, 27
 - callShot, 27
 - connectToServer, 28
 - enterSetupPhase, 29
 - gameOver, 30
 - getMainThreadEventHandler, 30
 - getSetupPanel, 31
 - handleGameEvent, 31
 - handleQuitGameEvent, 32
 - init, 33
 - playerReady, 34
 - quitGame, 34
 - sendEmote, 35
 - showEmote, 36
 - showError, 36
 - startGame, 37
- GameEvent, 38
 - GameEvent, 39
 - hit, 39
 - hitShip, 39

- nextPlayerId, 39
- playerId, 39
- position, 40
- ServerResponse.h, 131
- sunk, 40
- GameInstance, 40
 - executeShot, 40
 - getGameState, 41
 - isReady, 42
 - joinGame, 42
 - reset, 43
 - startGame, 44
- GameOver
 - AudioPlayer, 7
- gameOver
 - GameController, 30
 - GameState, 50
- GameOverEvent, 45
 - GameOverEvent, 46
 - ServerResponse.h, 131
 - winnerPlayerId, 46
- GameState, 46
 - addPlayer, 48
 - addShips, 49
 - ClientState, 48
 - finish, 49
 - Finished, 47
 - gameOver, 50
 - GameState, 48
 - getCurrentPlayerId, 50
 - getOppShipSunk, 51
 - getOtherPlayer, 51
 - getPlayer, 52
 - getPlayerGrid, 52
 - getPlayers, 52
 - getShip, 53
 - getState, 53
 - getWinner, 53
 - Playing, 47
 - registerShot, 54
 - removePlayer, 55
 - ServerState, 48
 - shotsIsLegal, 55
 - start, 56
 - Starting, 47
 - State, 47
 - Type, 47
 - updateBoards, 57
 - updateOppShipSunk, 58
- GameWindow, 59
 - GameWindow, 59
 - onClose, 60
 - setStatus, 60
 - showPanel, 60
- generateRandomUuid
 - uuid, 107
- getCurrentPlayerId
 - GameState, 50
- getEmote
 - SendEmote, 86
- getGameState
 - GameInstance, 41
- getGrid
 - SetupManager, 92
- getId
 - Player, 78
 - Ship, 99
- getImage
 - EmoteHandler, 22
- getImageLarge
 - EmoteHandler, 23
- getLength
 - Ship, 99
- getMainThreadEventHandler
 - GameController, 30
- getName
 - Player, 78
- getOppShipSunk
 - GameState, 51
- getOrientation
 - Ship, 100
- getOtherPlayer
 - GameState, 51
- getPlayer
 - GameState, 52
- getPlayerGrid
 - GameState, 52
- getPlayerId
 - ClientRequest, 15
- getPlayerName
 - JoinGame, 66
- getPlayers
 - GameState, 52
- getPosition
 - CallShot, 12
 - Ship, 100
- getReadyButton
 - SetupPanel, 95
- getReadyText
 - SetupPanel, 95
- getRequestType
 - ClientRequest, 15
- getServerAddress
 - ConnectionPanel, 17
- getServerPort
 - ConnectionPanel, 18
- getSetupPanel
 - GameController, 31
- getShip
 - GameState, 53
- getShipButton
 - SetupPanel, 96
- getShips
 - StartGame, 104
- getSound
 - EmoteHandler, 23

- getState
 - GameState, 53
- getUserName
 - ConnectionPanel, 18
- getValue
 - InputField, 64
- getWinner
 - GameState, 53
- GridType
 - ViewGrid, 108
- handleGameEvent
 - GameController, 31
- handleQuitGameEvent
 - GameController, 32
- handleRequest
 - RequestHandler, 83
- hasSunken
 - Ship, 100
- highlightTiles
 - PlacementGrid, 74
- Hit
 - AudioPlayer, 7
- hit
 - GameEvent, 39
 - Ship, 100
- hitShip
 - GameEvent, 39
- Horizontal
 - Ship, 99
- ImagePanel, 62
 - ImagePanel, 62
 - onSize, 63
 - paintEvent, 63
- init
 - ClientNetworkManager, 12
 - GameController, 33
- InputField, 64
 - getValue, 64
 - InputField, 64
- isReady
 - GameInstance, 42
- JoinGame, 65
 - ClientRequest.h, 125
 - getPlayerName, 66
 - JoinGame, 65
- joinGame
 - GameInstance, 42
- JoinGameSuccess, 66
 - JoinGameSuccess, 67
 - ServerResponse.h, 131
 - wasSuccessful, 67
- listenerLoop
 - ServerNetworkManager, 88
- LOG
 - Logger.h, 123
- log
 - Logger, 68
- Logger, 67
 - log, 68
 - setPrefix, 69
- Logger.h
 - LOG, 123
- main
 - main.cpp, 115
- main.cpp
 - main, 115
 - wxIMPLEMENT_APP, 114
- MainGamePanel, 70
 - buildGameState, 70
 - displayEmote, 71
 - MainGamePanel, 70
- MiddleFinger
 - EmoteEvent.h, 128
- Miss
 - AudioPlayer, 7
- Mocking
 - EmoteEvent.h, 128
- nextPlayerId
 - GameEvent, 39
- NLOHMANN_JSON_SERIALIZE_ENUM
 - serialization.h, 133
- onClose
 - GameWindow, 60
- onConnectButtonClicked
 - ConnectionPanel, 19
- OnInit
 - Battleship, 9
- OnKeyDown
 - SetupPanel, 96
- OnMouseClicked
 - PlacementGrid, 75
- OnMouseMotion
 - PlacementGrid, 76
- onPlayerLeft
 - ServerNetworkManager, 89
- OnReadyButtonClicked
 - SetupPanel, 97
- onSize
 - ImagePanel, 63
- operator<=>
 - Coordinate, 20
 - ServerResponse, 90
- operator()
 - std::hash< uuid >, 61
- operator==
 - Player, 79
 - uuid, 107
- opp
 - ViewGrid, 108
- Orientation
 - Ship, 99

- own
 - ViewGrid, 108
- paintEvent
 - ImagePanel, 63
- Panic
 - EmoteEvent.h, 128
- parseResponse
 - ClientNetworkManager, 13
- placedAllShips
 - SetupManager, 92
- PlacementGrid, 72
 - cellX_prev, 76
 - cellY_prev, 76
 - displayGrid, 73
 - highlightTiles, 74
 - OnMouseClicked, 75
 - OnMouseMotion, 76
 - PlacementGrid, 73
- PlaceShip
 - AudioPlayer, 7
- placeShip
 - SetupManager, 92
- play
 - AudioPlayer, 8
- PlayAgain, 77
 - ClientRequest.h, 125
 - PlayAgain, 77
- Player, 77
 - getId, 78
 - getName, 78
 - operator==, 79
 - Player, 78
- PlayerGrid, 79
 - PlayerGrid, 80
 - playerId, 80
 - shipsPlaced, 80
 - shotsFired, 80
 - shotsReceived, 80
- playerId
 - EmoteEvent, 22
 - GameEvent, 39
 - PlayerGrid, 80
- playerReady
 - GameController, 34
- players
 - StartGameSuccess, 105
- Playing
 - GameState, 47
- PopUp
 - AudioPlayer, 7
- position
 - GameEvent, 40
- QuitGame, 81
 - ClientRequest.h, 125
 - QuitGame, 81
- quitGame
 - GameController, 34
- QuitGameEvent, 82
 - QuitGameEvent, 82
 - quitPlayerId, 82
 - ServerResponse.h, 131
- quitPlayerId
 - QuitGameEvent, 82
- registerShot
 - GameState, 54
- removePlayer
 - GameState, 55
- RequestHandler, 83
 - handleRequest, 83
- RequestType
 - ClientRequest.h, 125
- reset
 - GameInstance, 43
- ResponseListenerThread, 84
 - Entry, 85
 - ResponseListenerThread, 85
- ResponseType
 - ServerResponse.h, 131
- responseType
 - ServerResponse, 90
- RussianWarshipGoFuckYourself
 - EmoteEvent.h, 128
- SelectShip
 - AudioPlayer, 7
- SendEmote, 86
 - ClientRequest.h, 125
 - getEmote, 86
 - SendEmote, 86
- sendEmote
 - GameController, 35
- sendRequest
 - ClientNetworkManager, 13
- serialization.h
 - from_json, 133
 - NLOHMANN_JSON_SERIALIZE_ENUM, 133
 - to_json, 133
- ServerNetworkManager, 87
 - ~ServerNetworkManager, 87
 - broadcastMessage, 88
 - listenerLoop, 88
 - onPlayerLeft, 89
 - ServerNetworkManager, 87
- ServerResponse, 89
 - ~ServerResponse, 90
 - operator<=>, 90
 - responseType, 90
 - ServerResponse, 90
- ServerResponse.h
 - EmoteEvent, 131
 - ErrorResponse, 131
 - GameEvent, 131
 - GameOverEvent, 131
 - JoinGameSuccess, 131
 - QuitGameEvent, 131

- ResponseType, 131
- StartGameSuccess, 131
- ServerState
 - GameState, 48
- setOrientation
 - Ship, 101
- setPosition
 - Ship, 101
- setPrefix
 - Logger, 69
- setStatus
 - GameWindow, 60
- SetupManager, 91
 - _selectedShip, 93
 - _ships_placed, 93
 - getGrid, 92
 - placedAllShips, 92
 - placeShip, 92
 - SetupManager, 91
- SetupPanel, 94
 - getReadyButton, 95
 - getReadyText, 95
 - getShipButton, 96
 - OnKeyDown, 96
 - OnReadyButtonClicked, 97
 - SetupPanel, 94
- Ship, 98
 - getId, 99
 - getLength, 99
 - getOrientation, 100
 - getPosition, 100
 - hasSunken, 100
 - hit, 100
 - Horizontal, 99
 - Orientation, 99
 - setOrientation, 101
 - setPosition, 101
 - Ship, 99
 - Vertical, 99
- ShipPanel, 102
 - ShipPanel, 102
 - update, 103
- shipsPlaced
 - PlayerGrid, 80
- shotsIsLegal
 - GameState, 55
- shotsFired
 - PlayerGrid, 80
- shotsReceived
 - PlayerGrid, 80
- showEmote
 - GameController, 36
- showError
 - GameController, 36
- showPanel
 - GameWindow, 60
- showShips
 - ViewGrid, 109
- showShots
 - ViewGrid, 109
- start
 - GameState, 56
- StartGame, 103
 - ClientRequest.h, 125
 - getShips, 104
 - StartGame, 104
- startGame
 - GameController, 37
 - GameInstance, 44
- StartGameSuccess, 105
 - players, 105
 - ServerResponse.h, 131
 - StartGameSuccess, 105
 - startingPlayerId, 106
- Starting
 - GameState, 47
- startingPlayerId
 - StartGameSuccess, 106
- State
 - GameState, 47
- std::hash< uuid >, 61
 - operator(), 61
- sunk
 - GameEvent, 40
- to_json
 - serialization.h, 133
- ToString
 - uuid, 107
- Type
 - GameState, 47
- update
 - ShipPanel, 103
- updateBoards
 - GameState, 57
- updateOppShipSunk
 - GameState, 58
- uuid, 106
 - generateRandomUuid, 107
 - operator==, 107
 - ToString, 107
 - uuid, 106
- Vertical
 - Ship, 99
- ViewGrid, 108
 - GridType, 108
 - opp, 108
 - own, 108
 - showShips, 109
 - showShots, 109
 - ViewGrid, 109
- wasSuccessful
 - JoinGameSuccess, 67
- what

BattleshipException, [10](#)
winnerPlayerId
GameOverEvent, [46](#)
wxIMPLEMENT_APP
main.cpp, [114](#)

x
Coordinate, [20](#)

y
Coordinate, [21](#)