Communication Protocol

Communication between the server and clients is all done through the serialization and deserialization of Event classes. Below we describe the different types of events that are used and their functions.

Event: This is the parent class of all the other Event types. It contains a “playerID” attribute which is set by the receiver *after* transmission of the Event over the network. This represents the ID of the player with which the Event is associated with.

ConnectEvent: The first message that the client sends to the server is a ConnectEvent. This is basically a request to the server for the client to join the game. The event also includes a boolean attribute which specifies if the client wants to be added as a spectator or an actual player.

ConnectAcceptedEvent: This message is sent back to the client from the server if the connection request was accepted.

ConnectRejectedEvent: This message is sent back to the client from the server if the connection request was rejected (ie no player slots left)

GameStartEvent: This message is sent to all the clients when the game is starting. This signals that the server is now read to begin accepting GameKeyEvents.

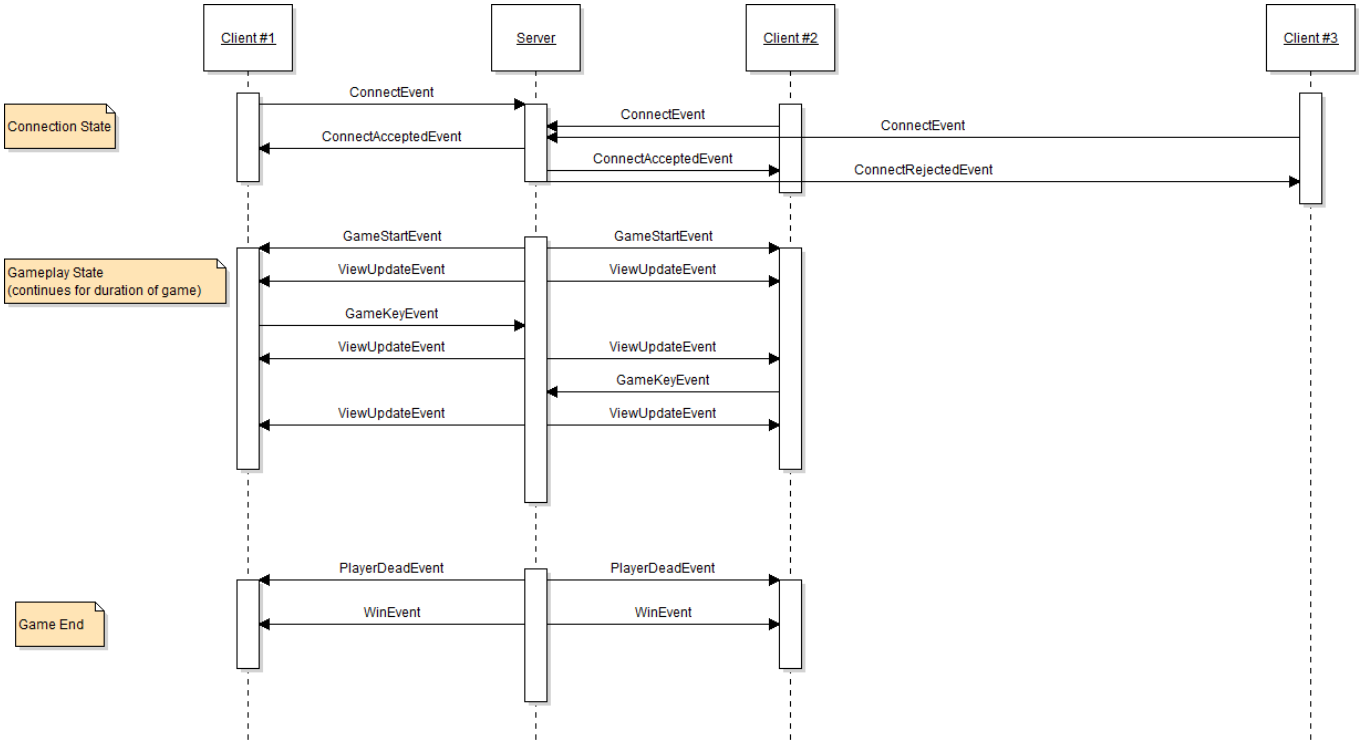
GameKeyEvent: This message is sent to the server from a client to signal that a button or command has been triggered on the client’s machine. For example if a player presses the Down button, a GameKeyEvent will be sent to the server, then the server will attempt to move that player downwards.

ViewUpdateEvent: This message is sent from the server to all the clients when a change has occurred to the model. It includes a serialized version of the Grid for the clients to render.

PlayerDeadEvent: This message is sent from the server to all the clients when a Player has died to signal that they cannot accept any more GameKeyEvents. The event includes a serialized version of the Player so that each client can tell if it was them that died.

WinEvent: This message is sent from the server to all the clients when a Player has won the game. This event includes a serialized version of the Player that won so that all clients can announce if it was them that won or lost. This also signals the end of the game.

See below for a sequence diagram of the communication that would happen during a typical game:


Client #1
Server
Client #2
Client #3
Connection State
ConnectEvent
ConnectEvent
ConnectEvent
ConnectAcceptedEvent
ConnectAcceptedEvent
ConnectRejectedEvent
Gameplay State
(continues for duration of game)
GameStartEvent
GameStartEvent
ViewUpdateEvent
ViewUpdateEvent
GameKeyEvent
ViewUpdateEvent
ViewUpdateEvent
GameKeyEvent
ViewUpdateEvent
ViewUpdateEvent
Game End
PlayerDeadEvent
PlayerDeadEvent
WinEvent
WinEvent