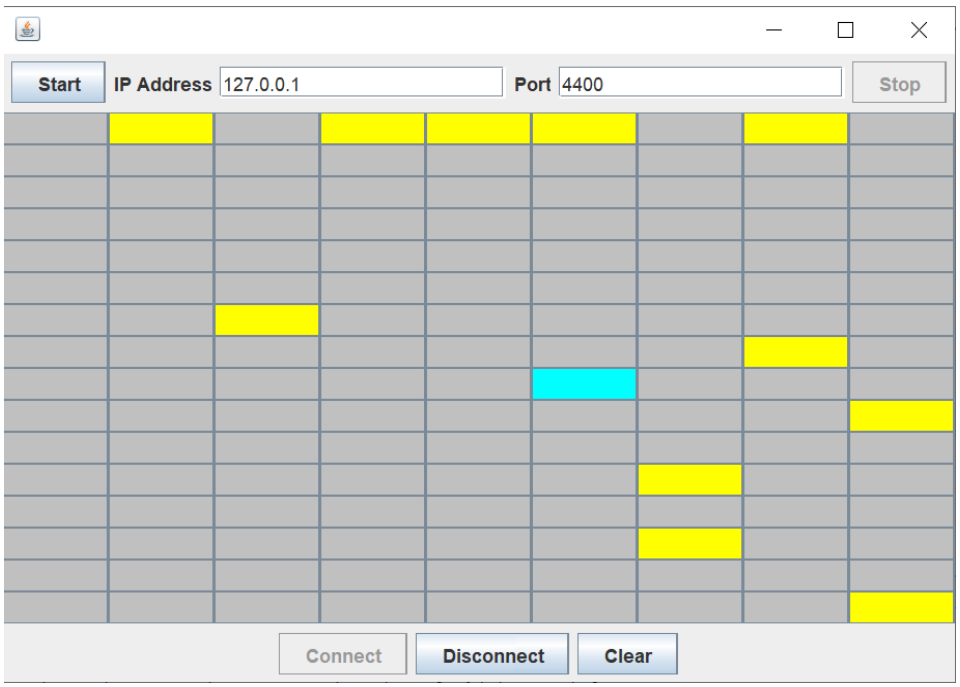
Programmazione Orientata Agli Oggetti

Si vuole realizzare un'applicazione client-server che permetta di giocare ad una variante di "Battaglia Navale". Lato client, l'utente sceglie dove posizionare le navi per poi avviare il "bombardamento" dal server. Il server sceglie la casella da colpire indicando la posizione sulla griglia. Il server è multithreading ed accetta connessioni da più client. Ogni client connesso gestisce una partita diversa (due client connessi allo stesso server di norma gestiscono partite differenti). I client una volta connessi possono iniziare a scaricare dal server stringhe contenenti informazioni riguardo all'indice della posizione nella scacchiera che il server vuole colpire. Le informazioni ricevute dal server avranno impatto sul client direttamente sull'interfaccia grafica. La durata massima di una partita è 21 secondi, nei quali il server "spara" 70 colpi. Tutte le stringhe sono inviate da client a server e viceversa utilizzando il carattere di fine linea come separatore.

Si richiede la realizzazione del client, interfaccia grafica e networking.

Il client dovrà essere composto da un frame che abbia per titolo *nome cognome e matricola* dello studente e che contenga due campi testuali (IP Address, Port), cinque pulsanti (Connect, Start, Stop, Disconnect, Clear) ed un pannello che a sua volta contenga 144 istanze di ColoredButton, disposte in una griglia 12*12.



All'avvio le caselle devono essere tutte del colore LIGHT_GRAY. L'utente deve poter cliccare le caselle e selezionare la casella (posizionamento delle navi). La selezione della casella comporta il cambio del colore della stessa in Color.CYAN. Il click su una casella selezionata riporta il colore a Color.LIGHT_GRAY. La gestione del click sulla casella è fornita già implementata nella

classe ColoredButton. Si consiglia l'utilizzo di tale classe fornita funzionante. Lo studente, tuttavia, è libero di ignorare o modificare la classe a piacimento qualora lo ritenga opportuno.

Per la disposizione degli elementi si faccia riferimento all'immagine.

NON è necessario implementare controlli non indicati nel testo (ad esempio chiusura finestra, interruzione partita in caso di sconfitta matematica).

Si implementi il seguente protocollo (12 punti):

- Alla pressione del pulsante "Connect", il client deve inviare una richiesta di connessione al server utilizzando IP e porta indicate nei campi testuali "IP Address" e "Port".
- Alla pressione del pulsante "Start", dopo la connessione, il client deve controllare che sia stata selezionata almeno una casella in griglia. In caso positivo, deve mandare il comando "start" al sever, il quale inizierà ad inviare ad intervalli regolari una stringa indicante la posizione_in_griglia, ovvero una stringa rappresentante un intero compreso tra 0 e 143. Se si considera la griglia come una matrice, il numero è calcolato seguendo la formula $i*12+j$ dove i,j sono rispettivamente gli indici di riga e colonna della matrice. Quindi ad esempio la cella [2][7] della matrice posizione_in_griglia = 2*12+7 = 31.

  NOTA BENE: Java aggiunge gli elementi alla griglia di un pannello utilizzando l'indice calcolato seguendo il meccanismo appena descritto. Le varie componenti grafiche offrono metodi per aggiungere gli elementi in posizioni specifiche. I riferimenti agli elementi, inoltre, possono essere gestiti in arrays (o matrici), si consiglia di utilizzare questa struttura dati per la gestione degli elementi in griglia.

  Il client alla ricezione della stringa dovrà cambiare colore alla casella nel pannello alla posizione indicata da posizione_griglia. Il colore diventerà Color.YELLOW nel caso in cui la casella non fosse stata selezionata ad inizio partita (colpo a vuoto), altrimenti il colore diventerà Color.GREEN (nave colpita). Per modificare il colore si suggerisce di utilizzare il metodo *public void changeColor(Color c)* della classe ColoredButton. Non è necessario gestire eventuali click in griglia dopo l'avvio della partita (pressione di Start). Si ipotizza che l'utente non prema le caselle a partite in corso.

  Alla ricezione di posizione_griglia = "-1", il client deve interrompere la ricezione dei messaggi e comunicare, tramite un messaggio in popup, il vincitore (Server o Utente). Il criterio di vittoria a fine partita è il seguente:

  if(n° delle caselle colorate Color.CYAN > 0) then User Wins;

  else Server Wins

  Al termine dello scambio, le tessere devono rimanere colorate. Nel caso in cui si avvii una nuova connessione dopo l'interruzione (si prema di nuovo Start), preliminarmente il colore di ogni tessera NON colorata di Color.CYAN deve essere riportato a Color.LIGHT_GRAY (come se si effettuasse una Clear, descritta in seguito).

- Alla pressione del pulsante "Stop", dopo l'avvio, il client deve mandare al server il comando "stop". Il server risponderà inviando una stringa che avrà posizione_griglia = "-1". Se la partita viene interrotta tramite "Stop", la vittoria deve essere forzatamente assegnata al Server. Deve essere possibile ricominciare successivamente una nuova partita semplicemente premendo Start.

- Alla pressione del pulsante “Disconnect” il client invia al server il comando “disconnect” e chiude la connessione.
- Alla pressione del pulsante “Clear”, il colore di ogni tessera deve essere riportato a Color.LIGHT_GRAY.

Gestire correttamente la possibilità di premere i pulsanti: in particolare

- all’avvio solo il pulsante “Connect” è abilitato
- una volta avviata una connessione, non deve essere premuto il pulsante “Connect”
- “Disconnect” può essere premuto solo se una connessione è avviata ma non si stanno ricevendo stringhe dal server
- “Stop” può essere premuto solo se il client è connesso e sta ricevendo stringhe
- “Start” può essere premuto solo se il client è connesso e non sta ricevendo stringhe
- “Clear” può essere premuto solo se non si stanno ricevendo stringhe dal server