Giulio Bombarda 16/05/2022

Documentazione e scelte progettuali per lo sviluppo di Jmemory

Durante il corso del progetto ho dovuto cambiare più volte le classi che mi permettevano di implementare le idee che avevo pensato, a causa della mia scarsa conoscenza della libreria Swing e dello strumento design di Netbeans che ho espanso informandomi sul web e discutendone con i compagni.

Ho ridimensionato il progetto per via delle tempistiche strette, a tal proposito ho rimosso la possibilità di inserire tessere con immagini personalizzate, la classe Mod e Gestore mod sono state rimosse e verranno implementate possibilmente alla fine del progetto.

Durante lo sviluppo del gioco ho cambiato più volte idea sulla progettazione cercando di trovare la soluzione più efficiente a tutti i problemi.

Come è strutturato il gioco

Partendo dalla parte di frontend, tutta la gestione della grafica è demandata alla classe MemoryGame, che contiene i form e i pannelli necessari al corretto utilizzo del gioco, ho deciso di separare da MemoryGame la classe Tessera, che rappresenta il pannello java (JPanel) della tessera del memory, dotata della possibilità di scoprirsi e coprirsi in base alla sollecitazione dell’evento Click.

Inclusi nella parte di frontend si possono trovare gli attributi di tipo Utente del giocatore e di un eventuale avversario per potersi interfacciare con la classe Partita del backend di cui parlerò più avanti.

Si trovano inoltre gli oggetti GestoreUtenti e GestoreFile, che si interfacciano rispettivamente con il login, la registrazione e con la scrittura del file di log, che serve a tenere traccia di tutte le operazioni effettuate del processo.

Per un corretto funzionamento dei metodi ho inserito anche due attributi di tipo intero: logMode e gameMode.

Nella classe MemoryGame sono stati implementati anche i gestori degli eventi che collaborano con dei metodi privati.

Il metodo principale che esegue il gioco e gestisce la partita si chiama initGame, sfrutta la classe Partita per gestire la ripartizione dei punti e la gestione delle tessere.

Parlando invece della parte di backend, la classe più importante che troviamo è Partita, contiene anch’essa gli oggetti di tipo Utente di utente e avversario per potersi sincronizzare con le classi del frontend, in più troviamo i contatori dei punti di entrambi giocatori, che verranno sfruttati in base alla modalità scelta dall’utente.

Oltre ad essi si trova l’ArrayList di tipo Tessera ha la funzione di contenere la collezione di tessere, caricate al momento dell’istanziazione dal metodo loadTessere, le immagini saranno contenute nella cartella tessere e verranno numerate da 0 al numero di tessere totali sottratto di 1.

La classe in questione necessiterà anche del metodo dedito a mischiare le tessere che verrà chiamato direttamente dalla funzione loadTessere quando avrà finito il processo di caricamento.

Il giocatore avrà la possibilità di giocare nelle tre modalità: multigiocatore, solitario e potrà sfidare il computer.

La classe, come anche le prossime contengono i vari metodi get e set oltre al override del metodo toString().

Un’altra classe importante che possiamo trovare è Utente, ha il compito di rappresentare l’utente del gioco per poi essere gestito dalla classe GestoreUtenti.

Ogni utente contiene un nickname, una password, un id univoco e il conteggio dei punti che è determinato in base alle partite vinte in multigiocatore.

La classe GestoreUtenti menzionata prima, si occupa della gestione degli utenti, per cui registrazione, controllo del login e possibilità di cambiare la password.

L’ultima classe che al momento è stata implementata è GestoreFile che si occupa della lettura e scrittura del file binario utenti.bin che funge da database per gli utenti.

GestoreFile lavora in concomitanza con la gestione degli utenti per poter garantire la corretta memorizzazione delle utenze del gioco.

GestoreFile si occupa infine di stampare all’interno del file di log denominato logs.log, un report su ogni azione che viene fatta specificandone data e ora dell’evento, il metodo printLog gestisce il file in modalità append.

Il diagramma UML delle classi verrà aggiornato il prima possibile delle ultime scelte progettuali effettuate.