Wordle 3.0

***Matteo Giorgi* 517183**

Il progetto consiste nella implementazione di Wordle, un gioco di parole web-based sviluppato da Josh Wardle nel 2021, acquistato poi dal New York Times a fine 2022.

Ogni 24h il gioco estrae casualmente dal proprio dizionario una Secret-Word di 5 lettere che il giocatore deve indovinare proponendo una Guessed-Word per ciascuno dei 6 tentativi massimi consentiti. Ad ogni tentativo, Wordle risponderà con indizi utili riguardo le lettere che compongono la Guessed-Word così da aiutare il giocatore a indovinare la Secret-Word giornaliera.

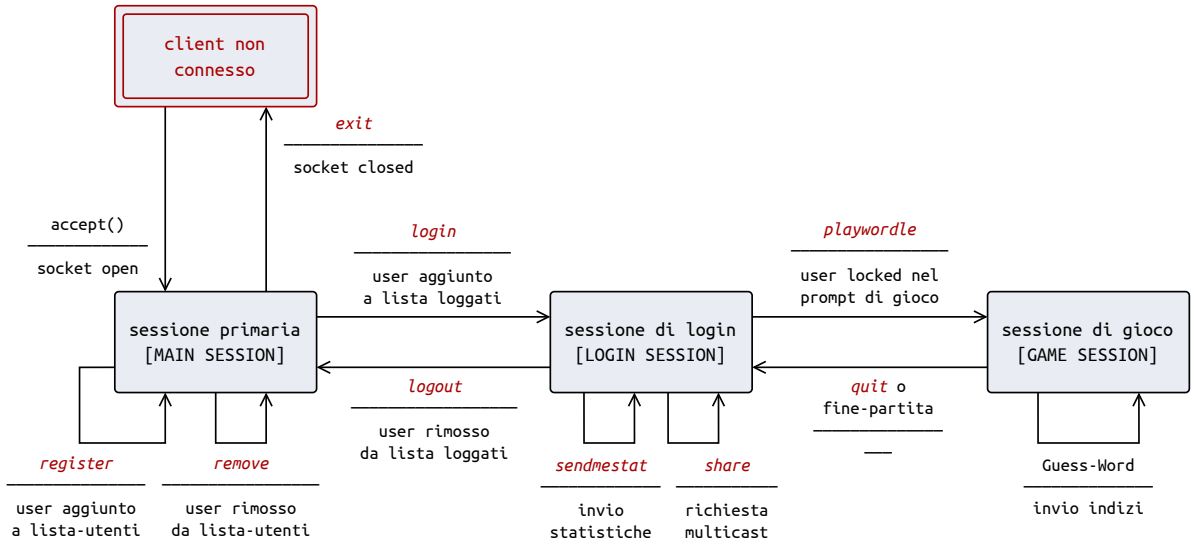
Questa implementazione consiste in una versione semplificata del gioco, che conserva la logica di base dell'originale ma apporta modifiche su alcune funzionalità come la condivisione social dei risultati (realizzata qui con un gruppo multicast), e l'assenza di una interfaccia grafica (sostituita da una semplice Command-Line UI).

La presente relazione è affiancata da una documentazione JavaDoc scritta come da specifica Oracle e dettagliata nelle linee guida delle Tecnical-Resources.

Wordle 3.0 usa una classica struttura client-server. Il server legge il proprio file di configurazione e si occupa di caricare in memoria l'elenco degli utenti, l'elenco delle parole (dizionario) e rimanere in attesa di connessioni su una welcome-socket (`ServerSocket`) appositamente allocata su una porta predefinita nel file di configurazione. Agganciato un client, il server lancierà dunque un nuovo Runnable (`Game`) con cui verranno soddisfatte le richieste, per poi rimettersi in attesa di una nuova connessione. Il client invece, dopo la lettura del proprio file di configurazione, ha l'unico scopo di connettersi al server con una socket (`Socket`) e inviare comandi sottoforma di *lines* (stringhe terminanti con il carattere di line-break).

Struttura del Progetto

Prima di entrare nelle specifiche dell'implementazione ecco qua sotto l'ASF che illustra i possibili stati di un client nelle varie fasi di gioco (si consideri ovviamente che client e server abbiano superato la fase iniziale di setup).



Modelli principali

- `Word` e `WordList` gestiscono le parole del gioco: la classe `Word` rappresenta una singola parola, mentre `WordList` il dizionario usato da *Wordle* per estrarre ciclicamente la *Secret-Word* e controllare la validità delle *Guessed-Words* inserite dall'utente in gioco.
- `User` e `UserList` in modo analogo alle classi precedenti rappresentano rispettivamente il singolo utente e l'insieme degli utenti registrati sul server del gioco.

Funzionalità del server

- `ServerSetup` e `ServerMain` sono le classi che descrivono le proprietà e identificano il punto di ingresso principale del server.
- `Game` è la classe responsabile della sessione di gioco per ciascun client, secondo i quattro stati specificati nell'ASF.
- `MulticastSender` gestisce la condivisione dei risultati attraverso un gruppo multicast.

Funzionalità del client

- `ClientSetup` e `ClientMain` analogamente alle loro controparti, rappresentano proprietà punto di ingresso principale del client.
- `MulticastReceiver` gestisce la condivisione dei risultati attraverso un gruppo multicast.

Classe `Word`

Classe che rappresenta la parola che i giocatori devono indovinare, è implementata usando due variabili private.

- *`currentWord`*: `String` che identifica la *Secret-Word* corrente.
- *`userSet`*: `Set<String>` che contiene i nomi degli utenti che hanno già giocato con la *Secret-Word* corrente. Un utente che avesse già giocato la *Secret-Word* corrente e chiedesse di iniziare una nuova partita, rimarrebbe in `[LOGIN SESSION]` (si veda AST).

Il costruttore della classe prende come parametro una `String` che rappresenta la parola da indovinare; non è consentito creare una parola *null*, causerebbe il lancio di una `IllegalArgumentException`.

Oltre ai metodi *`getWord`*, *`containsUser`* e *`addUser`*, la classe contiene anche *`getMask`*, necessario a `Game` per fornire informazioni al client sulla correttezza della *Guessed-Word* inserita, sottoforma di una maschera di caratteri speciali (`X`, `+`, `?`) come da specifica.

Classe `WordList`

Classe che rappresenta il vocabolario utilizzato da *Wordle* per estrarre la *Secret-Word* e controllare la validità delle *Guessed-Word* inserite dagli utenti durante una partita. Di seguito la struttura base.

- *`wordVocabulary`*: `List<String>` che contiene le parole del vocabolario.
- *`currentWord`*: istanza della classe `Word` che rappresenta la parola attualmente selezionata come *Secret-Word*.
- *`wordExtractor`*: `ScheduledExecutorService` che estrae una parola casuale dal vocabolario a intervalli regolari di tempo.
- *`extractWord`*: `Runnable` che rappresenta il task di estrazione casuale della *`currentWord`* dal *`wordVocabulary`*.

La classe `WordList` quindi fornisce la logica per gestire il vocabolario del gioco, selezionare una nuova parola segreta a intervalli regolari e verificare le supposizioni dell'utente.

Questa classe svolge un ruolo fondamentale nel gioco, poiché fornisce la meccanica di base per la selezione e la validazione delle parole.

Classe `User`

Classe `UserList`

Classe `ServerSetup`

Classe `ServerMain`

Classe `Game`

Classe `MulticastSender`

Classe `ClientSetup`

Classe `ClientMain`

Classe `MulticastReceiver`