Relazione Progetto Wordle

Leonardo Puosi

Mat: 628393

1 Schema Generale

1.1 Struttura

Il progetto "Wordle" è un'applicazione client-server che consente agli utenti di giocare tentando di indovinare una parola segreta.
L'applicazione è divisa in due package principali: "Server" e "Client", ognuno dei quali contiene il codice sorgente e i file di configurazione necessari per il corretto funzionamento del progetto.

Il **Server** è composto da:

- ServerMain.java
- ServerThread.java
- User.java
- WordExtractor.java.
- PersistanceHandler.java.

All'interno della directory per il Server è presente una sotto-directory "file" contenente:

- server.properties
- utenti.json
- words.txt

Il **Client** è composto da:

- ClientMain.java
- ClientThread.java

1.2 Fasi

Il progetto Wordle può essere suddiviso in tre fasi principali che definiscono il flusso di interazione dell'utente con il sistema. Di seguito, viene fornita una visione generale di queste tre fasi con le azioni possibili in ogni fase:

1. **Fase di Login/Registrazione**:
    - **Registrazione Utente**: Gli utenti hanno la possibilità di registrare un nuovo account fornendo un nome utente e una password. Il client invia queste informazioni al server, che le verifica e, se valide, crea un nuovo account utente.
    - **Login Utente**: Gli utenti già precedentemente registrati al sistema, possono effettuare l'accesso al loro account inserendo il nome utente e la password. Il server verifica le credenziali e, se corrette, permette all'utente di accedere al gioco.
2. **Fase del Menù Principale**:
    - **Inizia a Giocare**: Se l'utente non ha già inviato tentativi per la parola corrente, accede alla terza fase.
    - **Visualizza Statistiche**: L'utente riceve dal server le statistiche aggiornate all'ultima partita effettuata.
    - **Condividi Statistiche**: Il server condivide con tutti gli utenti online in quel momento l'esito dell'ultima partita dell'utente.
    - **Visualizza Altri Utenti**: L'utente visualizza le informazioni condivise dagli altri utenti tramite la funzione "Condividi Statistiche".
    - **Effettua il Logout**: L'utente si disconnette dal Server e torna alla fase 1.
3. **Fase di Gioco**:
    - **Invia la Parola**: Se un utente sceglie di iniziare una partita, entra nella fase di gioco Wordle. Qui può tentare di indovinare la parola e ricevere consigli sulla parola segreta.
    - **Torna al menù principale**: Gli utenti hanno anche la possibilità di uscire dalla fase di gioco in qualsiasi momento e tornare al menù principale. Il server consente agli utenti di interrompere il gioco attuale e tornare al menu principale senza perdere le statistiche rinunciando tuttavia ai tentativi rimasti per la parola corrente.

Queste tre fasi compongono il ciclo principale di interazione dell'utente con il sistema Wordle. Il sistema gestisce in modo efficace l'accesso degli utenti, le loro attività nel menù principale e le sessioni di gioco, garantendo un'esperienza di gioco fluida e interattiva.

2 Client

Il **Client** ha la funzione di interfaccia tra Utente e Server, limitandosi a gestire la comunicazione, inoltrando messaggi tra le due parti e traducendo le azioni e le richieste dell'utente in comandi comprensibili dal Server.
Di seguito sono elencate le azioni principali:

- Caricamento dei parametri modificabili dall'utente dal file client.properties presente nella sotto-directory "file".
- Instaurazione di una connessione TCP con il Server.
- Presentazione del menù di registrazione all'utente e attesa della selezione.
- Invio delle credenziali al server per la validazione e se corrette, permette di proseguire. In caso contrario fornisce all'utente la possibilità di riprovare.
- Creazione e avvio di un Thread (ClientThread) per la ricezione di pacchetti UDP:
  - Partecipazione al gruppo Multicast
  - Attesa dei messaggi
  - Salvataggio dei messaggi ricevuti in una struttura accessibile al Client-Main.
- Presentazione all'utente del menù principale e attesa della selezione della prossima azione.
- Invio della selezione effettuata dall'utente al Server e attesa della risposta.
- La risposta può essere una richiesta di input dal giocatore ("Input-request"), un messaggio da mostrare al giocatore, oppure una stringa riservata al Client per il passaggio da fase in fase (Es: "quit", "next").
- Il Client rimane in questa fase finchè l'utente non decide di effettuare il Logout.
- Una volta notificato il Server, il Client termina il thread in ascolto dei pacchetti UDP, svuota la struttura dati dedicata alle statistiche degli altri giocatori e torna alla prima fase.

3 Server

Il **Server** si occupa della gestione delle credenziali di accesso, delle informazioni degli utenti, della logica di gioco, della condivisione delle informazioni tra gli Utenti e della persistenza dei dati dopo ogni riavvio.

3.1 ServerMain

ServerMain è la classe principale del gioco Wordle. Di seguito sono elencate le azioni principali:

- Caricamento dei parametri di configurazione dal file server.properties.
- Ripristino dello stato precedente del server dal file JSON definito precedentemente utilizzando la libreria Gson e creando un "User" per ogni utente presente sul file. L'oggetto User viene memorizzato in una ConcurrentHashMap in modo da garantire la consistenza dei dati.
- Caricamento delle parole valide dal file definito in server.properties con l'inserimento in un HashSet in modo da effettuare ricerche efficienti.
- Avvio di un Thread tramite Scheduled Executor (WordExtractor) per selezionare periodicamente una parola segreta dal vocabolario.
- Inizializzazione del server socket e delle variabili di stato.
- Creazione di un handler per la terminazione (PersistanceHandler) che verrà eseguito quando il Server riceve il segnale SIGINT (CTRL+C).
- Avvio di un nuovo Thread tramite Cached Thread Pool (ServerThread) per ogni client connesso.

3.2 User

La classe User contiene tutte le informazioni relative ad un utente del gioco Wordle. Di seguito sono elencate le azioni principali:

- Inizializzazione e gestione delle informazioni dell'utente, come nome utente, password, statistiche di gioco e stato online.
- Fornisce getter e setter per l'accesso e la modifica delle informazioni dell'utente.

3.3 ServerThread

La classe ServerThread gestisce la comunicazione tra il Server e un Client. Di seguito sono elencate le azioni principali:

- Gestione dell'autenticazione e della registrazione degli utenti.

- Gestione delle partite degli utenti:
  - Ricezione delle parole inviate dagli utenti, valutazione e creazione dell'indizio da inviare in risposta.
  - Aggiornamento e invio delle statistiche del giocatore.
  - Condivisione a tutti gli utenti connessi, dell'ultima partita del giocatore attraverso un gruppo Multicast.
- Gestione della procedura di logout.

3.4 WordExtractor

La classe WordExtractor gestisce la selezione periodica di una parola segreta dal vocabolario. Di seguito sono elencate le azioni principali:

- Riceve il set di parole (vocabulary) e un riferimento alla parola segreta (SecretWord) da condividere con le istanze di ServerThread.
- Utilizza un generatore casuale per selezionare casualmente una parola dal vocabolario e la imposta come parola segreta.
- Aggiorna un contatore di ID di gioco (idgame) per garantire l'unicità di ogni partita.

3.5 PersistanceHandler

La classe PersistanceHandler gestisce la persistenza dei dati del server al momento della chiusura e la terminazione dei Thread avviati. Di seguito sono elencate le azioni principali:

- Gestione della chiusura pulita del server, inclusa la terminazione dei thread e la chiusura delle connessioni.
- Salvataggio dei dati degli utenti, inclusi username, password e statistiche di gioco, in un file JSON utilizzando la libreria Gson.

4 Strutture Dati Utilizzate

4.1 Client

- *ConcurrentLinkedQueue(String)* **playerStats**: Utilizzata per il salvataggio dei messaggi ricevuti tramite connessione UDP.

4.2 Server

- *AtomicReference(String)* **SecretWord**: Contiene la parola che gli utenti devono indovinare, il thread WordExtractor seleziona la nuova parola e modifica la stringa atomicamente.
- *HashSet(String)* **vocabulary**: Contiene l'insieme di parole che sono considerate valide, viene inizializzato all'avvio del server e permette di fare ricerche in tempi brevi evitando di accedere spesso al file.
- *AtomicInteger* **idgame**: Contiene l'id del gioco, un valore che identifica univocamente ogni sessione di gioco. La sessione inzia con la selezione della parola da indovinare e termina al momento del cambio della parola.
- *ConcurrentHashMap(String, User)* **login**: Contiene le informazioni relative ad ogni utente.
- *AtomicBoolean* **online**: Presente all'interno della classe User e rappresenta lo stato di un giocatore, se è connesso al server in un determinato istante il valore sarà true, altrimenti conterrà false.

5 Thread Attivati

5.1 Client

Il client utilizza 2 thread:

1. ClientMain: Avviato 1 sola volta
2. ClientHandler: Avviato 1 sola volta

5.2 Server

Il server utilizza 4 o più thread:

1. ServerMain: Avviato 1 sola volta
2. WordExtractor: Avviato periodicamente per la procedura di cambio parola.
3. PersistanceHandler: Avviato alla ricezione del segnale di terminazione.
4. ServerThread: Viene avviata un'istanza ogni volta che un Client si connette.

6 Istruzioni per l'uso

6.1 Compilazione manuale

Per **compilare** il Server è sufficiente eseguire nella directory "Progetto" il comando:

javac -cp ".:Server/lib/gson-2.10.1.jar" Server/*.java

Per avviare il **Server** si può utilizzare il comando:

java -cp ".:Server/lib/gson-2.10.1.jar" Server.ServerMain

Per **compilare** il Client è sufficiente eseguire nella directory "Progetto" il comando:

javac Client/*.java

Per avviare il **Client** si può utilizzare il comando:

java Client.ClientMain

6.2 Makefile

È possibile compilare ed eseguire Client e Server utilizzando un makefile.
Per **compilare** l'intero progetto è sufficiente eseguire nella directory del progetto il comando:

make

Per avviare il **Server** si può ulilizzare il comando:

make Server

Per avviare il **Client** si può utilizzare il comando:

make Client