Metodi Avanzati Di Programmazione

(Documentazione Progetto)

Nome progetto: AdventureCastleGame

Autori: Vito Giuseppe Natale, Giuseppe Piemontese, Cristina Tedeschi.

Sommario

1. Descrizione generale del caso di studio
2. Diagramma delle classi con annessa descrizione
3. Specifica algebrica di una struttura dati
4. Descrizione di come abbiamo utilizzato i vari argomenti del corso

* File
* Jdbc
* Lambda expression
* Thread
* Swing

DESCRIZIONE GENERALE DEL CASO DI STUDIO

Il caso di studio che abbiamo scelto riguardava la creazione di un’avventura testuale da sviluppare in java.

Su queste direttive abbiamo sviluppato una storia ambientata in un maniero abbandonato dove il giocatore è impegnato ad addentrarsi in questo maniero per riuscire a salvare la sua innamorata tenuta prigioniera da un maniaco chiamato Mister X che l’ha rapita. L’impresa non è così semplice perché Mister X si è rinchiuso in una stanza segreta del maniero e tramite degli scagnozzi è riuscito a nascondere le chiavi che permettono di raggiungerlo. Il compito del giocatore è quello di recuperare la giusta chiave, di trovare MisterX e un modo per sconfiggerlo così da liberare la sua amata.

Il giocatore si deve avventurare nel maniero ed entrare nelle stanze che gli ispirano di piu per cercare le chiavi e soprattutto trovare la stanza dove è rinchiusa la sua amata, ogni stanza è composta da una descrizione che permette al giocatore di capire che azioni svolgere, inoltre in ogni stanza ci sono degli oggetti che possono essere utili per il completamento delle azioni al fine di ottenere le chiavi.

DESCRIZIONE DELLE COMPONENTI GRAFICHE:

All’avvio del gioco si apre una finestra dove il giocatore può effettuare il login inserendo il suo username di gioco, se ha già giocato in passato c’è la possibilità di recuperare (se desiderato dal giocatore) i dati di gioco della sessione precedente come gli oggetti che aveva nell’inventario e la posizione in cui si trovava, altrimenti viene registrato come nuovo utente e inizia una nuova partita.

Al termine di questa funzione si apre la schermata di gioco composta da un pannello scorrevole dove appariranno tutte le descrizioni comprese i comandi inseriti dall’giocatore, quest’ultimi verranno inseriti in una barra di testo situata al di sotto del pannello scorrevole, sul lato sinistro ci sono 3 pulsanti :

Mappa1: questo permette di visualizzare una piantina del primo piano del maniero, pensata per facilitare l’orientamento del giocatore;

Mappa2: questo pulsante invece fa visualizzare la piantina del secondo piano del maniero;

Help: questo pulsante apre una finestra dove al suo interno ci sono i comandi che il giocatore può utilizzare e che vengono accettati dal gioco.

Un'altra componente grafica è situata all’interno della stanza biblioteca dove compare un jFrame che mette alla prova le capacità intellettuali del giocatore con 3 indovinelli.

La finestra è composta da un pannello centrale dove vengono visualizzati gli indovinelli e una barra sotto al pannello che cliccandoci escono svariati titoli di libri selezionabili, il giocatore deve indovinare quale titolo del libro corrisponde alla risposta dell’indovinello.

Al completamento di questi 3 indovinelli viene rilasciata una chiave come ricompensa per la bravura mostrata nel risolvere gli indovinelli.

DIAGRAMMA DELLE CLASSI FONDAMENTALI



Il programma parte da Grafica e crea un’istanza di Engine passandogli AdventureCastleGame, in seguito invoca il metodo start di Engine. Da Engine viene creato un nuovo oggetto StartGame passandogli l’oggetto engine di Grafica. StartGame si occupa di fare il login dell’utente. Una volta eseguito il login restituisce il controllo ad Engine. Quest’ultimo comunica con la classe grafica per tutta la durata dell’esecuzione.

Quando viene inserito un comando nella JTextFild (writer) e viene premuto il tasto invio si attiva un evento actionPerformed che al suo interno chiama il metodo execute della classe Engine. Questo metodo analizza tramite il parser la stringa passata in input e se riconosce il comando richiama il metodo nextMove di AdventureCastleGame che in base al comando svolge delle azione che vengono riportate sulla grafica di Grafica.

SPECIFICA ALGEBRICA (STRING)

**Specifica Sintattica**

sorts: String, Char, Int, Bool;

**operations**

new() → String;

add(String, Char) → String;

append(String, String) → String;

length(String) → Int;

 isEmpty(String) → Bool;

equal(String, String) → Bool;

|  |  |  |
| --- | --- | --- |
| Osservatori di s’ | Costruttori di s’ | |
|  | New() | Add(s,c) |
| Lenght(s’) | 0 | Lenght(s)+1 |
| isEmpty(s’) | True | false |
| Append(p,s’) | p | Add (append(p,s) , c) |

Operatore equal(String, String) → Bool; new()

|  |  |  |
| --- | --- | --- |
| Costruttori di t’ | Costruttori | |
| New() | Add (p,c) |
| New() | true | false |
| New() | false | if r=c then equal(t,p)  else false |

**Specifica Semantica**

declare: s, t, p : String, c: Char

length(new()) = 0;

length(add(s, c)) = length(s) + 1;

isEmpty(new()) = true;

isEmpty(add(s, c)) = false;

append(p, new()) = p;

append(p, add(t, c)) = add(append(p, t), c);

equal(new(), new()) = true;

equal(add(p, c), new()) = false;

equal(new(), add(t, c)) = false;

equal(add(t, c), add(p, c)) = equal(t, p)

DESCRIZIONE DEGLI ARGOMENTI DEL CORSO USATI NEL PROGETTO

**LAMBDA EXPRESSION: priva LAMBDA EXPRESSION:**

**Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente**

nella classe AdventureCastleGame vi è il metodo showObjects che prende in input una lista di oggetti. Da questa lista viene ricavato uno stream su cui vengono applicate le operazioni di filter, map e forEach allo scopo di ottenere i singoli elementi nella lista in “output”.

 UTILIZZO DEL FILE:

Immagine che contiene testo, schermata, numero, Carattere

Descrizione generata automaticamente

Questo metodo viene creato per leggere da file, scannerizza due righe alla volta:

-la prima contiene un valore intero che viene utilizzato come indice dell’array;

-la seconda contiene una stringa che corrisponde alla descrizione di una stanza.

Inoltre, abbiamo utilizzato il metodo chiamato writeString che si occupa di scrivere sul file ma viene usato solo nel caso si vogliono aggiungere nuove stanze con nuove descrizioni al gioco.

UTILIZZO SWING:

Grafica.java:

questa classe crea una grafica per il gioco dove abbiamo posizionato le componenti al suo interno in un GridLayout, si può interagire con ogni componente della grafica essendo che gli abbiamo aggiunto ad ogni componente la gestione di un evento.

StartGame.java:

questa classe crea una finestra che permette all’utente di effettuare l’accesso o registrarsi, è stata creata come la precedente e quindi abbiamo gestito la situazione alla pari.

IndovinellIU.java:

Le librerie swing le abbiamo utilizzate per creare la finestra degli indovinelli.

Con le librerie Swing di Java abbiamo creato una classe che estende JFrame che è il componente principale dove poi all'interno nel GridLayout (che è un tipo di Layout che ti permette di posizionare i vari elementi come in una griglia) abbiamo posizionato gli altri componenti, tra cui un JTextArea, una JComboBox, un JLabel e un pulsante. Abbiamo aggiunto anche gli ActionListener che catturano gli eventi dei vari componenti, tra cui la selezione di uno dei libri nel JComboBox che controlla se la risposta è esatta oppure no. Nel caso di risposta esatta passa all'indovinello successivo.

UTILIZZO DEI THREAD

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Abbiamo pensato di utilizzare un thread che gestisce la chiusura del gioco. Quest’ultimo da la possibilità di chiamare i metodi che salvano i dati di gioco del giocatore, attende 5 secondi e termina l’esecuzione del gioco.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Questo thread inizializza l’attacco del nemico, e aspetta ogni 5 secondi per far partire un altro attacco, cosi via finché il nemico non muore.

JDBC:

public Connection connectToDatabase() throws SQLException

Questo metodo si trova nella classe DatabaseManagement e si occupa di stabilire una connessione con il database, una volta impostate le credenziali e creato l’url di connessione apre una connessione con il database.

public boolean isUsernameExists() throws SQLException

Questo metodo richiama la connessione al database ed effetua una query per verificare che i nome utente inserito esista già, ne caso esiste ritorna il valore true.