Documentazione Progetto JUno

Sapienza - Teledidattica
Facoltà di Informatica
Corso di Metodologie di Programmazione
Cosmo Vellucci 1760067

Ultimo aggiornamento in data 14/10/2022

SAPIENZA
UNIVERSITÀ DI ROMA

Sommario

Finalità della documentazione

Questo documento ha lo scopo di presentare l'ideazione, la creazione, lo sviluppo e informazioni del progetto **JUno** per il corso di Metodologie di Programmazione (Sapienza – Unitelma), le cui specifiche sono presenti nel seguente PDF.

MDP 21-22 - progetto_ver_0.8_be

L'unico partecipante al progetto è lo stesso che ha scritto la documentazione, ovvero Cosmo Vellucci.

Ambiente di test e sviluppo

Il progetto è stato interamente sviluppato e testato su sistema operativo Windows 10, usando l'IDE gratuito Eclipse. Per quest'ultimo non sono stati utilizzati plug-in.

Applicazioni coinvolte nello sviluppo

Le applicazioni coinvolte durante lo sviluppo del progetto sono stati i seguenti:

- **GIMP**, per la creazione e la modifica degli asset
- **Eclipse IDE**, per la creazione del codice
- **Microsoft Word**, per la stesura di questa documentazione
- **Draw.io**, per disegnare la mappa degli stati e il diagramma UML

Implementazione delle regole di gioco

Le regole di gioco implementate sono quelle classiche, che riassumiamo di seguito:

- L'obbiettivo del partecipante è scartare le carte compatibili fino a quando si rimane senza carte
- Se si rimane senza carte si calcola il punteggio di fine round. Si vince quando un giocatore possiede 500 punti.
- Il punteggio è calcolato come riportato nel manuale di gioco

Invece non sono stati implementati i seguenti aspetti

- Carte personalizzate
- Carte jolly baratto
- Meccanica di sfida (vale a dire che la carta *JOLLY PESCA4* può essere giocata normalmente, senza alcuna restrizione)

Per maggiori informazioni, si faccia riferimento alla documentazione ufficiale allegata di seguito

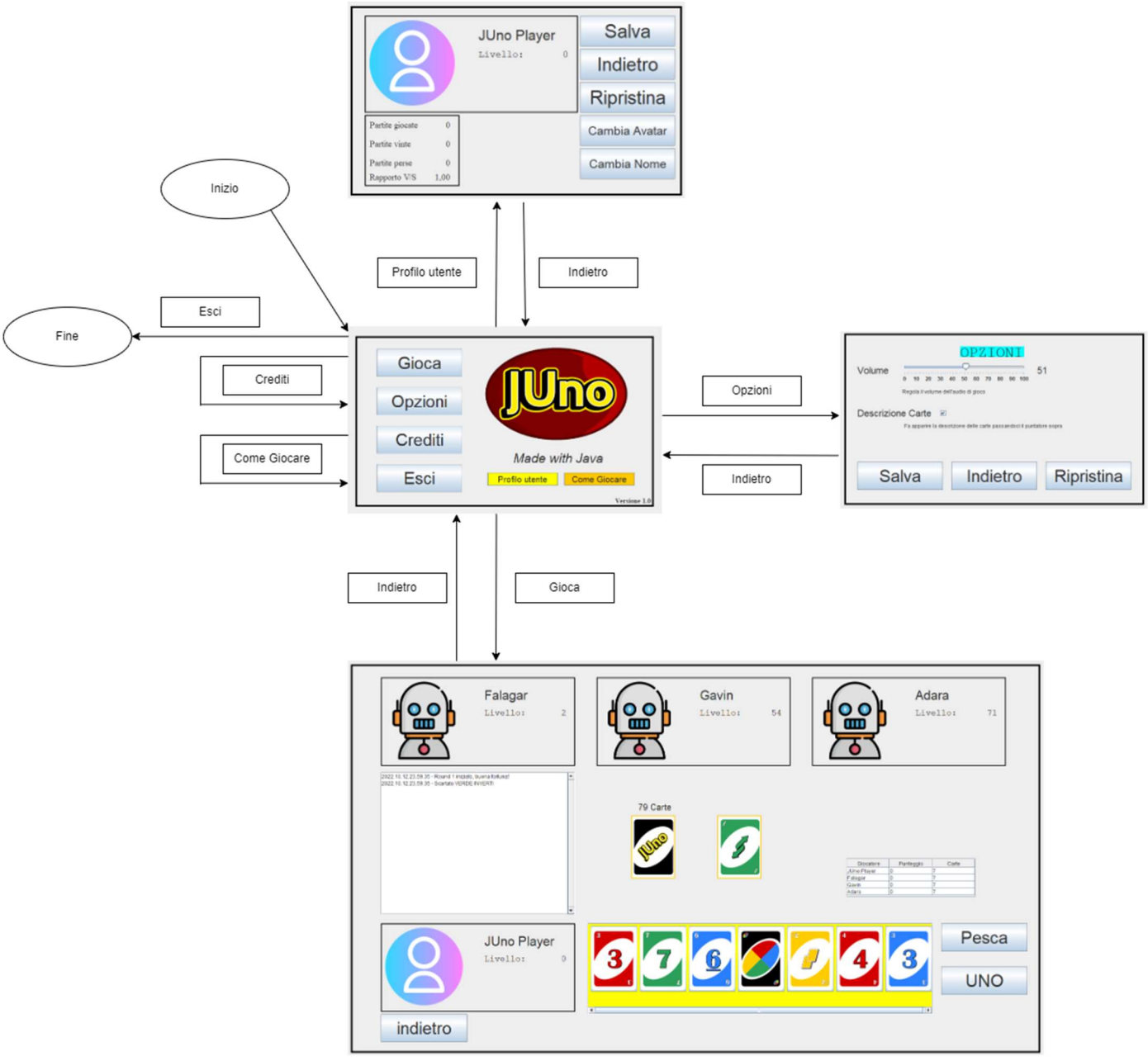
regole-uno-da-scaricare-pdf.pdf

Mappa degli stati

Per facilitare la comprensione dell'interfaccia, nell'immagine sottostante viene mostrata una mappa degli stati del programma.

In particolare:

- Gli archi indicano una transizione di stato dell'interfaccia
- I rettangoli rappresentano il click del bottone relativo
- Cliccando sul pulsante "Crediti" viene aperta una finestra di dialogo che mostra i crediti di gioco.
- Cliccando sul pulsante "Come Giocare" viene aperto il manuale delle regole di UNO (lo stesso riportato nel paragrafo *implementazione delle regole di gioco*), tramite il programma di default per aprire PDF impostato sul PC sul quale viene avviato il gioco.



Scelte Implementative

Un frame, diversi pattern

Il programma inizializza un solo frame, e su di esso si applicano diversi panel (si passa da un panel all'altro rimuovendo il precedente e aggiungendo quello nuovo.

File di salvataggio

Siccome è necessario memorizzare localmente informazioni relative alle impostazioni e alle statistiche del giocatore, i file di salvataggio e profilo utente sono stati realizzati in formato binario (e non plaintext) al fine evitare manomissione da parte dell'utente.

Pattern singleton

Il pattern singleton è stato applicato alle seguenti classi:

- MainFrame.java - perché il gioco deve essere avviato una sola volta
- ModelOpzioni.java - perché le opzioni sono singolarmente relative a una sola istanza del gioco
- AudioPlayer.java - perché ci deve essere un solo audio player per la singola istanza del gioco

Inoltre, MainFrame.java presenta dei campi e metodi statici, dato che è il punto principale sul quale appoggiare diversi panel che vanno aggiunti e rimossi.
Infine, la classe ModelOpzioni.java è statica perché deve essere accessibile da più classi, e la classe AudioPlayer.java fornisce dei metodi statici perché viene chiamata appunto ogni volta che è necessario riprodurre un file audio (come un certo bottone oppure durante la partita).

Pattern MVC

Dato che per la realizzazione del progetto è necessario un'interfaccia grafica, si è deciso di usare il pattern MVC per facilitare la scrittura logica del codice. Le classi coinvolte in questo pattern sono i seguenti:

- Opzioni (ControllerOpzioni.java, ModelOpzioni.java, ViewOpzioni.java)
- Partita (ControllerPartita.java, ModelPartita.java, ModelProfiloUtente.java, ViewPartita.java, ViewProfiloutente.java)
- Utente (ControllerProfiloUtente.java, ModelProfiloUtente.java, ViewProfiloUtente.java)

Descrizione carte

Nel progetto si è voluto simulare una feature di accessibilità che si fa spesso nei videogiochi moderni: nelle opzioni è possibile attivare la funzionalità *descrizione carte*, che permette al passaggio del mouse su una carta di mostrare tramite testo il suo colore e il suo numero.

Stream adottati

Gli stream adottati nel progetto sono riportati qui di seguito:

- Classe ControllerPartita.java
    - Funzione passaTurno
        - Viene usato uno stream per calcolare la prima carta giocabile dalla mano di un NPC
    - Funzione calcolaPunteggio
        - Viene usato uno stream per calcolare velocemente il punteggio di fine round