
Cluedo
Righi Michele
Sarneri Enrico
Righi Lorenzo

Cos'è Cluedo?

- Applicativo dell'omonimo *gioco da tavolo*
- Permette di effettuare partite in *Giocatore Singolo* (offline) e *Multigiocatore* (online)
- Offre agli utenti un'esperienza di gioco *avvincente* ed *equilibrata*

Punti focali

- Obiettivi e requisiti
- Pattern architetturali
- Principi di design
- Pattern di design
- Ambiente di sviluppo e tecnologie
- Revisione e testing
- Conclusione

Obiettivi e requisiti

- Usabilità

  Interfacce grafiche progettate per essere il più semplici ed intuitive possibili

- Performance

  Tempi di risposta brevi ed efficiente utilizzo delle risorse hardware

- Sicurezza

  File di log per tenere traccia delle operazioni effettuate all'interno del sistema e protocollo TLS per le comunicazioni online

Pattern architetturali

- Client/Server su 3 livelli

  Poiché l'applicazione prevede una modalità multigiocatore, si è scelta come architettura il Client/Server: un *UtenteProprietario* ospita una Partita e gli altri *Utenti* possono partecipare accedendovi tramite un registro remoto, dopo essersi autenticati. Il terzo livello è quello necessario per la persistenza dei dati degli utenti.

- Model-View-Presenter (MVP)

  Abbiamo scelto questo pattern perché permette di separare modello del dominio, logica di presentazione dei dati e logica di business. Inolte, ci permette di avere una migliore organizzazione dei package e lo abbiamo preferito rispetto all'MVC in quanto il model non interagisce direttamente con la view.

Principi di Design

- Dependency Inversion Principle

  Applicato nelle interfacce dei controller al fine di rendere tutte le funzionalità estendibili tramite la dipendenza da astrazioni stabili

- Single Responsability Principle

  Applicato nella classe LogWriter separando la funzionalità di Log da quelle proprie del gioco.

- Open-Closed Principle

  Applicato nella classe di dominio Filter, al fine di rendere possibile l'aggiunta di nuovi filtri senza modificare l'interfaccia Filter.

Pattern di Design

- Pattern Adapter
- Pattern Singleton
- Pattern Strategy
- Pattern State

Pattern Adapter

Problema:

Necessità di convertire l'interfaccia "originale" della classe Sala, in una con funzionalità aggiuntive: UtenteProprietario si aspetta di poter svolgere alcune funzionalità in più (da superutente) che riguardano l'amministrazione della sala e delle sue proprietà.

Soluzione:

Convertire l'interfaccia originale di Sala nell'interfaccia diversa di SalaProprietario, tramite l'utilizzo di una classe che funge da adattatore. Così facendo è possibile riutilizzare la prima aggiungendovi nuove funzionalità.

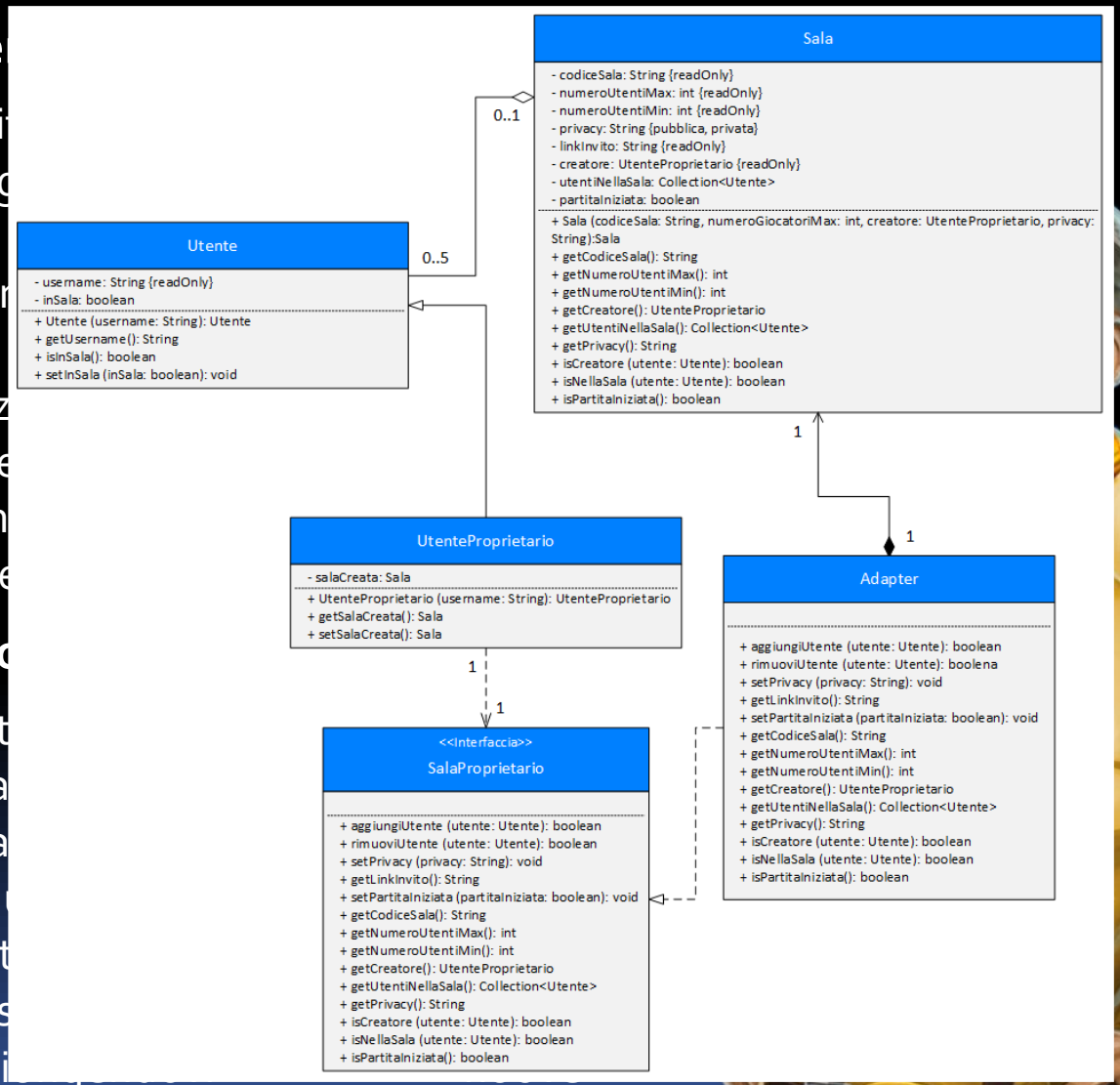
Pattern Adapter

Proble

Necessi
"orig
una
Uter
di
funz
supe
l'am
delle

Soluzio

Convert
Sala
Sala
di
adat
poss
aggi
funzionalità.



Pattern Singleton

Problema:

Necessità di avere la garanzia che durante l'esecuzione possa esistere al più un'istanza della classe Partita. Di conseguenza, sviluppando l'applicazione secondo il pattern MVC, si ha che anche il controller relativo (GestionePartitaController) deve avere una sola istanza.

Soluzione:

Il pattern Singleton permette di ottenere l'istanza della classe solo tramite un metodo statico getIstanza() (il costruttore è privato), al cui interno viene controllato se esiste già l'istanza. In caso contrario viene creata.

Pattern Singleton

Problema:

Necessità di avere
durante l'esec
esistere al più
classe Partita. I
sviluppando
secondo il patter
anche il con
(GestionePartita(
avere una sola ist

Soluzione:

Il pattern Singleto
ottenere l'istanza
tramite un r
getIstanza() (il
privato), al cu
controllato se es
In caso contrario

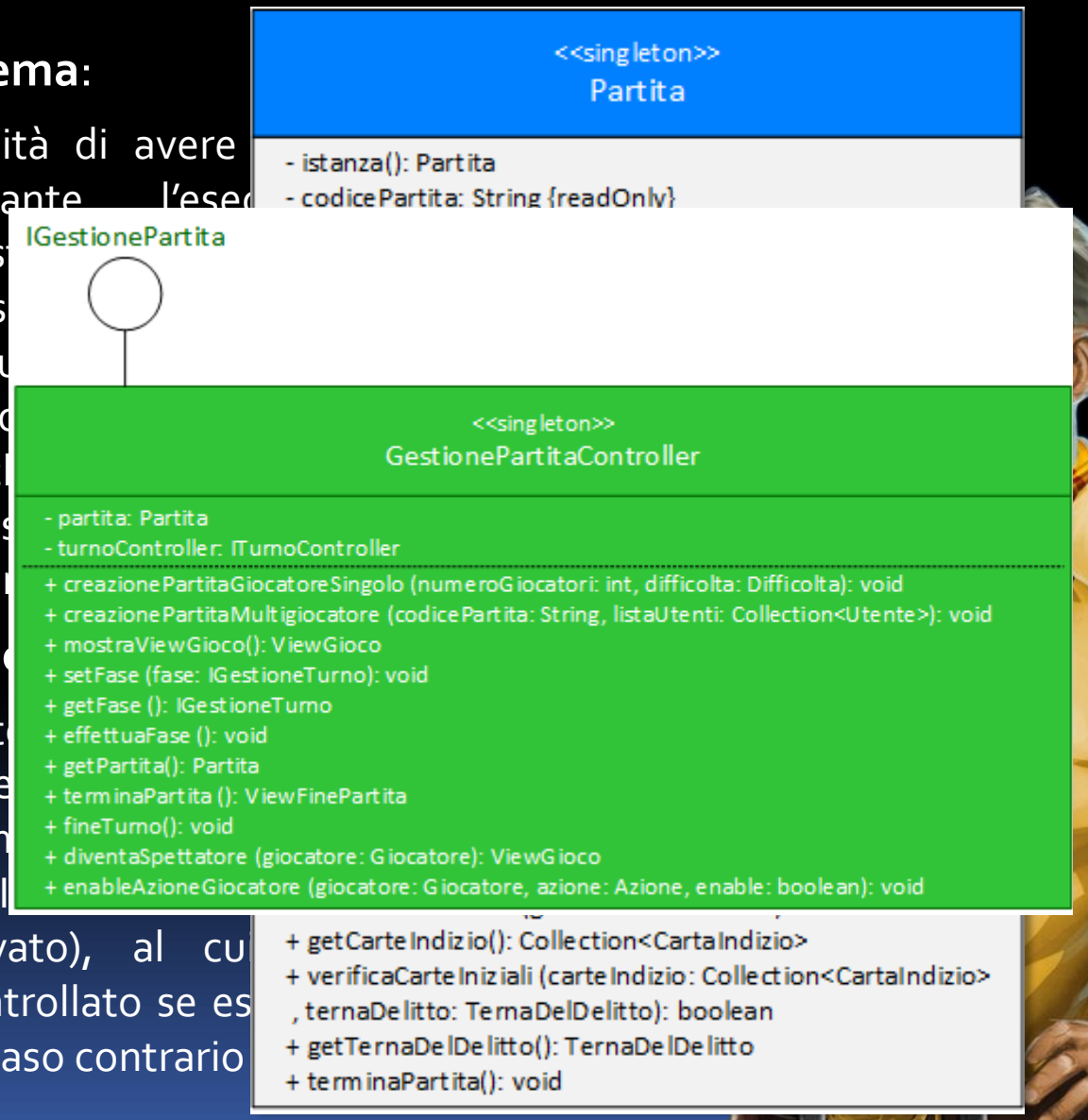
Pattern Singleton

Problema:

Necessità di avere
durante l'esec
esist
clas
svilu
secc
ancl
(Ges
aver

Soluzi

Il patt
otte
tran
getI
privato), al cu
controllato se es
In caso contrario



Pattern Strategy

Problema:

Trovare un metodo per filtrare le entry nel Gestore della Sicurezza secondo algoritmi differenti, che sia stabile ed estendibile (in quanto i Log sono l'unico metodo di analisi del sistema), con l'obbiettivo di riuscire ad applicare il Principio di Apertura/Chiusura (OCP).

Soluzione:

Il pattern Strategy permette di incapsulare algoritmi differenti in classi che implementano la stessa interfaccia. Così facendo queste vengono viste dall'esterno nello stesso modo e diventano intercambiabili.

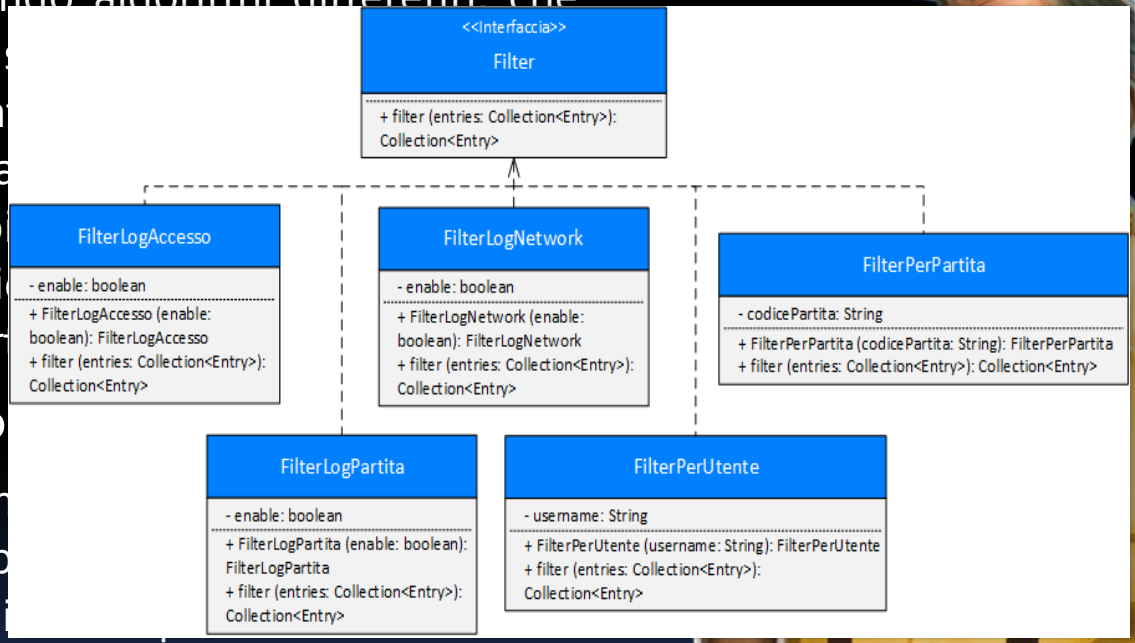
Pattern Strategy

Problema:

Trovare un metodo per filtrare le entry nel Gestore della Sicurezza secondo algoritmi differenti che sia
quant
di a
l'obb
appli
Aper

Soluzio

Il pattern
incap
classi
interfaccia. Così facendo queste vengono viste dall'esterno nello stesso modo e diventano intercambiabili.



Pattern State

Problema:

Necessità di suddividere il comportamento di un Giocatore in base alla fase in cui si trova e permettere il passaggio da una fase all'altra secondo una certa logica di transizione.

Soluzione:

Il pattern State permette ad un oggetto di cambiare il proprio comportamento a tempo di esecuzione in base allo *stato* in cui si trova, quindi è perfetto: nel nostro caso lo stato diventa la fase del Giocatore e le logiche di transizione vengono incapsulate nei vari metodi di esecuzione della fase.

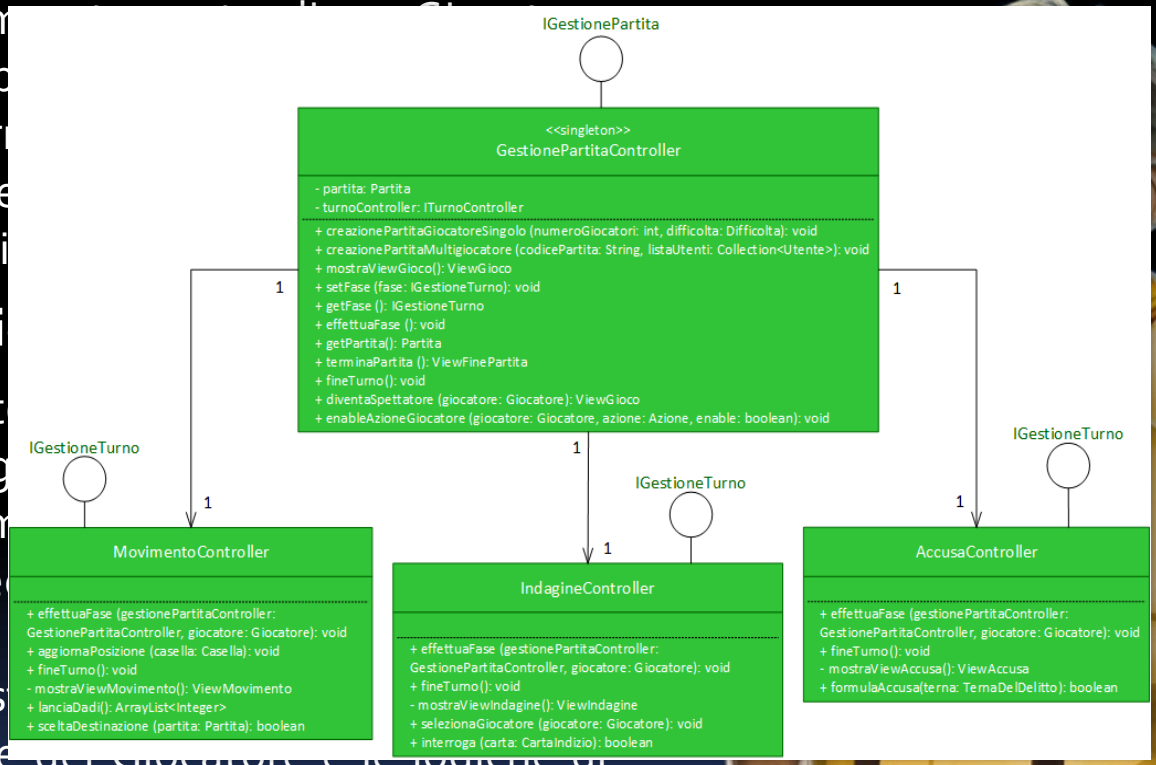
Pattern State

Problema:

Necessità di suddividere il com
in b
per
fase
logi

Soluzi

Il patt
ogg
com
ese
cui
nos
fase del Giocatore e le logiche di transizione vengono incapsulate nei vari metodi di esecuzione della fase.



Ambiente di Sviluppo e Tecnologie

- Eclipse e SceneBuilder
- Java (JDK 10.0)
- Java Virtual Machine (JVM)
- JavaFX SDK (10.0.2)
- JavaFX CSS

Ambiente di Sviluppo e Tecnologie

- Eclipse e SceneBuilder

  Come ambiente di sviluppo si è scelto *Eclipse* soprattutto per un fattore di comodità nella gestione e nell'organizzazione delle risorse del progetto. *SceneBuilder* semplifica il lavoro della parte grafica e la creazione delle varie View (file .fxml) dell'applicazione. I file così ottenuti vengono poi caricati direttamente nel codice Java.

- Java (JDK 10.0)

  E' il linguaggio orientato agli oggetti utilizzato per la scrittura del codice dell'applicazione. In particolare, la versione del *Java Development Kit* è la 10.0.2. Permette di compilare il codice sorgente scritto in linguaggio Java, in bytecode.

- Java Virtual Machine (JVM)

  E' l'ambiente di esecuzione delle applicazioni Java: esegue il codice del programma tradotto in bytecode, indipendentemente dall'architettura hardware su cui ci si trova.

Ambiente di Sviluppo e Tecnologie

- JavaFX SDK (10.0.2)

  JavaFX è la libreria utilizzata per la costruzione e l'utilizzo dei componenti grafici, che vengono mostrati all'utente tramite l'interfaccia grafica e con cui è possibile interagire.

- JavaFX CSS

  JavaFX CSS è il linguaggio utilizzato per definire la formattazione dei componenti grafici realizzati con la libreria JavaFX.

Revisione e Testing

L'applicazione realizzata è suddivisa in molteplici moduli come previsto nel documento di progetto, la modularità è risultata utile sia per lo sviluppo in sé che per il testing. Oltre a un primo collaudo generale finalizzato alla verifica del funzionamento di costruttori e della correttezza dei parametri che era stato indicato nel documento abbiamo effettuato ulteriori test atti a verificare il corretto funzionamento dei singoli metodi e dei vari oggetti grafici interattivi, a tal fine ogni componente identificato da un'interfaccia grafica è stato fatto eseguire singolarmente tramite funzioni main locali.

Conclusione

- Cluedo è un'applicazione semplice e facile da utilizzare, che rispetta pienamente i requisiti forniti dal committente.
- Il prototipo mostra una versione base dell'applicativo e permette di:
  - navigare liberamente all'interno del Menu;
  - accedere alle funzionalità offline (modificare le impostazioni, visualizzare le regole ed accedere al tutorial);
  - giocare una partita in modalità Giocatore Singolo, contro avversari virtuali;
  - visualizzare ed interagire coi log.
- Sviluppi futuri
  - modalità multigiocatore;
  - possibilità di scegliere la difficoltà degli avversari virtuali;
  - aggiunta di un sistema di punteggio con classifica;
  - Sviluppo di una versione per browser web (con compatibilità multipiattaforma).

Buona visione!