

Progettazione object-oriented di un'interfaccia grafica JavaFX per il simulatore Alchemist

Tesi in Programmazione ad Oggetti

Niccolò Maltoni
0000719734

Alma Mater Studiorum - Università di Bologna
Campus di Cesena

Outline

1 Introduzione

2 Alchemist

- Introduzione ad Alchemist
- Il modello di Alchemist

3 L'interfaccia classica

4 L'interfaccia implementata

Introduzione

Lo scopo di questa tesi è la progettazione e la successiva implementazione di un'interfaccia grafica 2D per il simulatore *Alchemist*.

Introduzione

Lo scopo di questa tesi è la progettazione e la successiva implementazione di un'interfaccia grafica 2D per il simulatore *Alchemist*.

La nuova interfaccia permette di interagire con la simulazione a tempo di esecuzione e di vedere chiaramente rappresentate informazioni su di essa.

Introduzione

In particolare, è supportata una *struttura modulare di effetti* che rende facilmente osservabili determinate entità del sistema ed eventuali loro proprietà:

Introduzione

In particolare, è supportata una *struttura modulare di effetti* che rende facilmente osservabili determinate entità del sistema ed eventuali loro proprietà:

- l'effetto non fa più riferimento al singolo nodo, bensì costituisce una funzione dall'intero ambiente alla rappresentazione grafica.

Introduzione

In particolare, è supportata una *struttura modulare di effetti* che rende facilmente osservabili determinate entità del sistema ed eventuali loro proprietà:

- l'effetto non fa più riferimento al singolo nodo, bensì costituisce una funzione dall'intero ambiente alla rappresentazione grafica.
- gli *stack* di effetti realizzati possono essere serializzati su file di testo in formato JSON.

Introduzione

Si è scelto di mantenere un'interfaccia il più possibile *user-friendly*, mantenendo un design più simile ai simulatori a scopo videoludico per favorire l'utilizzo da parte di utenti inesperti.

Introduzione

Si è scelto di mantenere un'interfaccia il più possibile *user-friendly*, mantenendo un design più simile ai simulatori a scopo videoludico per favorire l'utilizzo da parte di utenti inesperti.

Lo stile estetico al quale si è deciso di allinearsi è il *Material Design* di Google¹ e la libreria grafica utilizzata per l'implementazione è stata *JavaFX*.

¹<https://material.io>

Outline

1 Introduzione

2 Alchemist

- Introduzione ad Alchemist
- Il modello di Alchemist

3 L'interfaccia classica

4 L'interfaccia implementata

Alchemist

Introduzione ad Alchemist

Alchemist² è un meta-simulatore estendibile completamente open-source che esegue su Java Virtual Machine (JVM), nato all'interno dell'Università di Bologna e reperibile su GitHub³.

²<http://alchemistsimulator.github.io>

³<https://github.com/AlchemistSimulator/Alchemist>

Alchemist

Introduzione ad Alchemist

Alchemist² è un meta-simulatore estendibile completamente open-source che esegue su Java Virtual Machine (JVM), nato all'interno dell'Università di Bologna e reperibile su GitHub³.

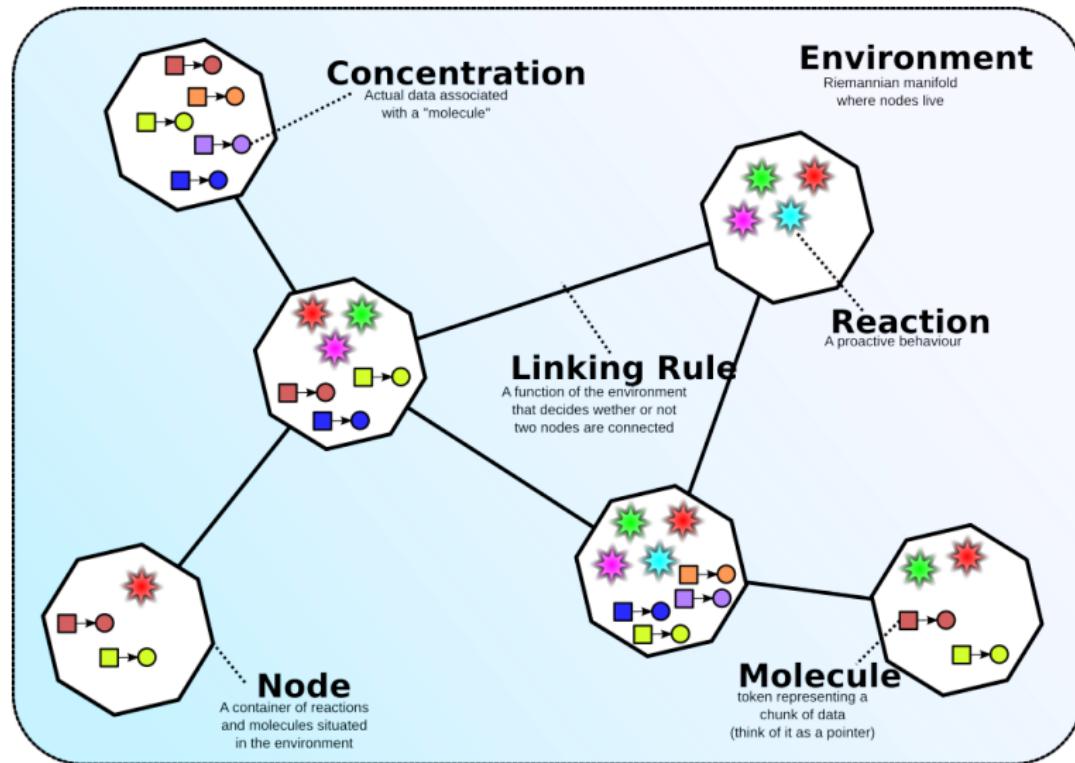
L'idea dietro al progetto è quello di riuscire ad avere un framework di simulazione il più possibile generico, in grado di simulare sistemi di tipologia e complessità diverse, mantenendo le prestazioni dei simulatori non generici (come ad esempio quelli impiegati in ambito chimico).

²<http://alchemistsimulator.github.io>

³<https://github.com/AlchemistSimulator/Alchemist>

Alchemist

Il modello di Alchemist



Alchemist

Il modello di Alchemist

Molecola Una *Molecola* rappresenta il nome dato ad un particolare dato all'interno di un *Nodo*, del quale ne astrae parte dello stato.

Alchemist

Il modello di Alchemist

Molecola Una *Molecola* rappresenta il nome dato ad un particolare dato all'interno di un *Nodo*, del quale ne astrae parte dello stato.

Concentrazione La *Concentrazione* di una *Molecola* è il valore associato alla proprietà rappresentata dalla *Molecola*.

Alchemist

Il modello di Alchemist

Molecola Una *Molecola* rappresenta il nome dato ad un particolare dato all'interno di un *Nodo*, del quale ne astrae parte dello stato.

Concentrazione La *Concentrazione* di una *Molecola* è il valore associato alla proprietà rappresentata dalla *Molecola*.

Nodo Il *Nodo* è un contenitore di *Molecole* e *Reazioni* che risiede all'interno di un *Ambiente* e che astrae una singola entità.

Alchemist

Il modello di Alchemist

Molecola Una *Molecola* rappresenta il nome dato ad un particolare dato all'interno di un *Nodo*, del quale ne astrae parte dello stato.

Concentrazione La *Concentrazione* di una *Molecola* è il valore associato alla proprietà rappresentata dalla *Molecola*.

Nodo Il *Nodo* è un contenitore di *Molecole* e *Reazioni* che risiede all'interno di un *Ambiente* e che astrae una singola entità.

Ambiente L'*Ambiente* è l'astrazione che rappresenta lo spazio nella simulazione ed è l'entità che contiene i *Nodi*.

Alchemist

Il modello di Alchemist

Molecola Una *Molecola* rappresenta il nome dato ad un particolare dato all'interno di un *Nodo*, del quale ne astrae parte dello stato.

Concentrazione La *Concentrazione* di una *Molecola* è il valore associato alla proprietà rappresentata dalla *Molecola*.

Nodo Il *Nodo* è un contenitore di *Molecole* e *Reazioni* che risiede all'interno di un *Ambiente* e che astrae una singola entità.

Ambiente L'*Ambiente* è l'astrazione che rappresenta lo spazio nella simulazione ed è l'entità che contiene i *Nodi*.

Regola di collegamento La *Regola di collegamento* è una funzione dello stato corrente dell'*Ambiente* che associa ad ogni *Nodo* un *Vicinato*.

Alchemist

Il modello di Alchemist

Vicinato Un *Vicinato* è un'entità costituita da un *Nodo* detto “centro” e da un insieme di altri *Nodi* (i “vicini”).

Alchemist

Il modello di Alchemist

Vicinato Un *Vicinato* è un'entità costituita da un *Nodo* detto “centro” e da un insieme di altri *Nodi* (i “vicini”).

Reazione Una *Reazione* è un insieme di *Condizioni* sullo stato del sistema che qualora dovessero risultare vere innescherebbero l'esecuzione di un insieme di *Azioni*.
Ogni *Nodo* è costituito da un insieme (anche vuoto) di *Reazioni*.

Alchemist

Il modello di Alchemist

Vicinato Un *Vicinato* è un'entità costituita da un *Nodo* detto “centro” e da un insieme di altri *Nodi* (i “vicini”).

Reazione Una *Reazione* è un insieme di *Condizioni* sullo stato del sistema che qualora dovessero risultare vere innescherebbero l'esecuzione di un insieme di *Azioni*.
Ogni *Nodo* è costituito da un insieme (anche vuoto) di *Reazioni*.

Condizione Una *Condizione* è una funzione che associa un valore numerico e un valore booleano allo stato corrente di un *Ambiente*.

Alchemist

Il modello di Alchemist

- Vicinato** Un *Vicinato* è un'entità costituita da un *Nodo* detto “centro” e da un insieme di altri *Nodi* (i “vicini”).
- Reazione** Una *Reazione* è un insieme di *Condizioni* sullo stato del sistema che qualora dovessero risultare vere innescherebbero l'esecuzione di un insieme di *Azioni*.
Ogni *Nodo* è costituito da un insieme (anche vuoto) di *Reazioni*.
- Condizione** Una *Condizione* è una funzione che associa un valore numerico e un valore booleano allo stato corrente di un *Ambiente*.
- Azione** Un'*Azione* è una procedura che provoca una modifica allo stato dell'*Ambiente*.

Outline

1 Introduzione

2 Alchemist

- Introduzione ad Alchemist
- Il modello di Alchemist

3 L'interfaccia classica

4 L'interfaccia implementata

L'interfaccia classica

L'architettura di Alchemist è progettata con paradigma *Model-View-Controller* (MVC), di conseguenza la suddivisione tra componente grafica (*View*) e il blocco “logico” composto da *Model* e *Controller* è netta.

L'interfaccia classica

L'architettura di Alchemist è progettata con paradigma *Model-View-Controller* (MVC), di conseguenza la suddivisione tra componente grafica (*View*) e il blocco “logico” composto da *Model* e *Controller* è netta.

Questa distinzione è evidente anche per quanto riguarda l'utilizzo pratico del software:

L'interfaccia classica

L'architettura di Alchemist è progettata con paradigma *Model-View-Controller* (MVC), di conseguenza la suddivisione tra componente grafica (*View*) e il blocco “logico” composto da *Model* e *Controller* è netta.

Questa distinzione è evidente anche per quanto riguarda l'utilizzo pratico del software:

- una simulazione su Alchemist può venire lanciata da terminale, senza che alcuna interfaccia grafica sia necessaria per tutta la durata del periodo di esecuzione ...

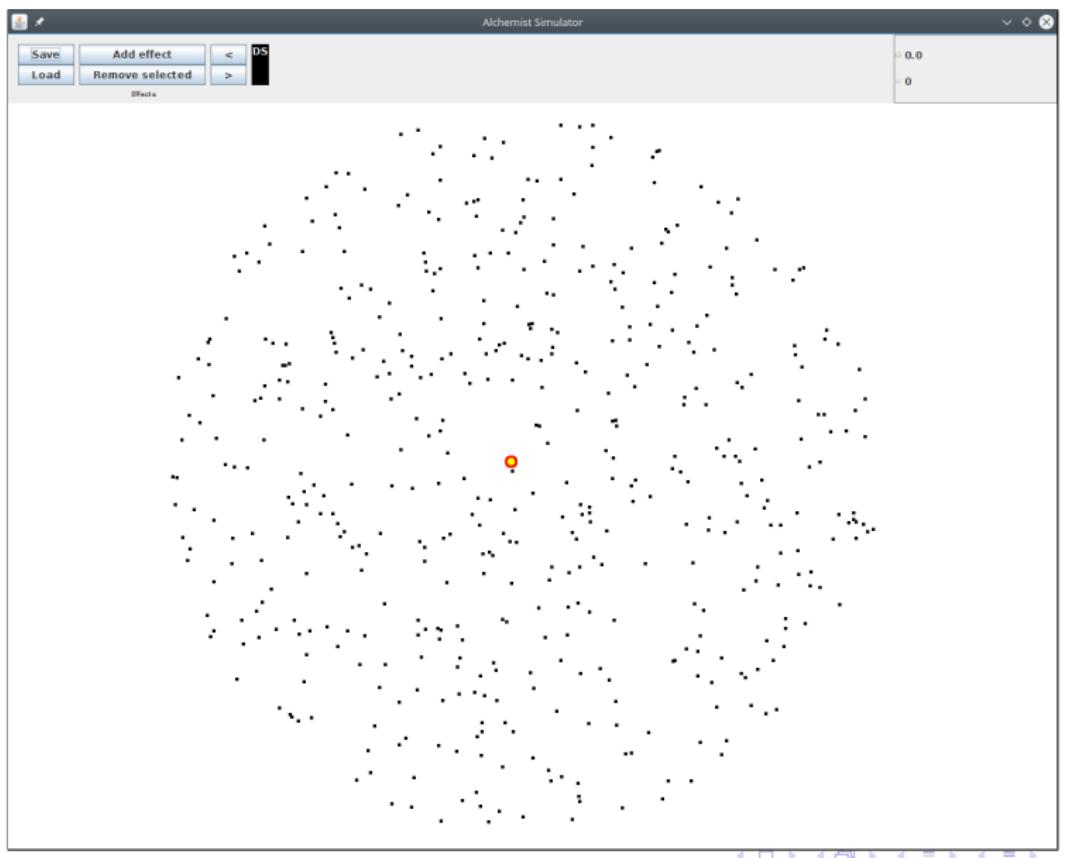
L'interfaccia classica

L'architettura di Alchemist è progettata con paradigma *Model-View-Controller* (MVC), di conseguenza la suddivisione tra componente grafica (*View*) e il blocco “logico” composto da *Model* e *Controller* è netta.

Questa distinzione è evidente anche per quanto riguarda l'utilizzo pratico del software:

- una simulazione su Alchemist può venire lanciata da terminale, senza che alcuna interfaccia grafica sia necessaria per tutta la durata del periodo di esecuzione ...
- ... oppure essere inizializzata, lanciata e controllata in tempo reale dalla sua interfaccia grafica.

L'interfaccia classica



L'interfaccia classica

L'interfaccia utente classica di Alchemist è caratterizzata da un'usabilità appena sufficiente, funzionale alle necessità di un utilizzatore esperto, ma non adeguata a fornire un'esperienza completa e *user-friendly* ad un utente "standard". In particolare:

L'interfaccia classica

L'interfaccia utente classica di Alchemist è caratterizzata da un'usabilità appena sufficiente, funzionale alle necessità di un utilizzatore esperto, ma non adeguata a fornire un'esperienza completa e *user-friendly* ad un utente "standard". In particolare:

- Il sistema di controllo non è intuitivo: non sono presenti bottoni di interazione per, ad esempio, avviare o fermare la simulazione o per cambiare la modalità di interazione con la zona in cui viene rappresentato l'ambiente, bensì scorciatoie da tastiera.

L'interfaccia classica

L'interfaccia utente classica di Alchemist è caratterizzata da un'usabilità appena sufficiente, funzionale alle necessità di un utilizzatore esperto, ma non adeguata a fornire un'esperienza completa e *user-friendly* ad un utente "standard". In particolare:

- Il sistema di controllo non è intuitivo: non sono presenti bottoni di interazione per, ad esempio, avviare o fermare la simulazione o per cambiare la modalità di interazione con la zona in cui viene rappresentato l'ambiente, bensì scorciatoie da tastiera.
- L'aspetto estetico è datato e non aderisce ad alcun design grafico conosciuto, né ne definisce uno proprio.

L'interfaccia classica

L'interfaccia utente classica di Alchemist è caratterizzata da un'usabilità appena sufficiente, funzionale alle necessità di un utilizzatore esperto, ma non adeguata a fornire un'esperienza completa e *user-friendly* ad un utente "standard". In particolare:

- Il sistema di controllo non è intuitivo: non sono presenti bottoni di interazione per, ad esempio, avviare o fermare la simulazione o per cambiare la modalità di interazione con la zona in cui viene rappresentato l'ambiente, bensì scorciatoie da tastiera.
- L'aspetto estetico è datato e non aderisce ad alcun design grafico conosciuto, né ne definisce uno proprio.
- Le capacità di rappresentazione, rappresentate dagli effetti, sono legate strettamente ai nodi e limitano la libertà di rappresentazione di ciò che avviene nella simulazione.

Outline

1 Introduzione

2 Alchemist

- Introduzione ad Alchemist
- Il modello di Alchemist

3 L'interfaccia classica

4 L'interfaccia implementata

L'interfaccia implementata

