FollowMe

1. Introduzione

Il progetto "FollowMe" è un'applicazione Java che offre la possibilità di simulare un gruppo di robot che si muovono in uno spazio virtuale. Questa simulazione è supportata da una libreria Java appositamente sviluppata per gestire il comportamento dei robot all'interno dell'ambiente simulato. È presente anche un'interfaccia grafica annessa che permette agli utenti di visualizzare in modo più chiaro l'ambiente circostante e i robot prima e dopo l'esecuzione del programma.

Per lo sviluppo del progetto, ho utilizzato Gradle come sistema di automazione della compilazione e gestione delle dipendenze. Inoltre, ho ritenuto opportuno utilizzare Java SE 17 (LTS) e JavaFX nella versione 20.0.1. La scelta di questa versione di Java garantirà una stabilità a lungo termine, in quanto tale versione verrà supportata per un periodo esteso di tempo.

2. Responsabilità assegnate

Prima di iniziare il processo di sviluppo, ho deliniato gli obiettivi del progetto che sono andato poi a sviluppare per chiarire le specifiche funzionalità da implementare e i requisiti chiave. Ho definito le seguenti responsabilità che ho poi sviluppato nelle classi specificate:

- *Caratteristiche dei robot e i relativi comportamenti:*
  Implementata dalla classe `Robot` che rappresenta un robot nello spazio e specifica le sue capacità di movimento e di interazione.

- *Specifica dei comandi possibili all'interno di un programma:*
  Tutte le classi che rappresentano i comandi di base implementano l'interfaccia `RobotCommand` (ovvero, `MoveCommand`, `MoveRandomCommand`, `SignalCommand`, `UnsignalCommand`, `FollowCommand`, `ContinueCommand`, `StopCommand`). Invece, le classi che rappresentano i comandi di selezione ed iterazione (`RepeatCommand`, `UntilCommand`, `DoForever`) estendono la classe astratta `AbstractIterator` che a sua volta implementa l'interfaccia `Iterator`. Quest'ultima è un'estensione dell'interfaccia `RobotCommand`.

- *Rappresentazione delle figure che compongono l'ambiente:*
  Questa responsabilità è assegnata alle classi `Circle` e `Rectangle`. Esse estendono la classe `GeometricShape` che a sua volta implementa l'interfaccia di base `Shape`. Tali classi rappresentano le possibili figure presenti nell'ambienti ed implementano alcuni metodi di utilità.

- *Gestione del flusso di esecuzione:*
  Dunque, l'inizializzazione dell'applicazione, la generazione dei robot, il caricamento dell'ambiente e del programma da eseguire. Tali funzionalità sono implementate nel `Referee`, esso funge da controller dell'applicazione.

- *Compilazione dei comandi del programma e dell'ambiente:*
  Il parsing del programma e dell'ambiente caricato avviene nella classe `Handler`. Esso, è una sorta di compilatore che verifica la correttezza sintattica, carica l'ambiente e il programma da eseguire. Tale classe, per il parsing dell'ambiente utilizza la classe di utilità `ShapeParser`.

- *Esecuzione del programma nello sciame:*

Una volta caricato il programma viene salvato all'interno della classe `Executor` che si occupa della corretta esecuzione del programma gestendo anche eventuali loop annidati. Tale classe, per l'esecuzione dei singoli comandi utilizza la classe di utilità `CommandExecutor`.

Per una miglior consistenza dei dati ho creato le seguenti classi che mi permettono di rappresentare in modo più specifico i vari tipi di dato: `Point`, `Direction`, `Signal`. Inoltre, per una miglior organizzazione e gestione delle responsabilità ho creato anche alcune classi di utilità: `DirectionCalculator`, `DistanceCalculator`, `RandomPointGenerator`, `CommandRow`.

3. Istruzioni per l'esecuzione

Come già specificato, il programma realizzato utilizza Gradle e Java. Dunque, per compilarlo basterà eseguire il comando da terminale **`gradle build`** e per eseguirlo il comando **`gradle run`**.

Una volta eseguito il comando run si otterrà una finestra (*Figura 1*) in cui verrà chiesto prima il numero di robot che si desiderano generare randomicamente nella simulazione. Successivamente, apparirà un'ulteriore finestra (*Figura 2*) che permetterà di selezionare il tempo da dedicare per l'esecuzione di ogni singolo comando.

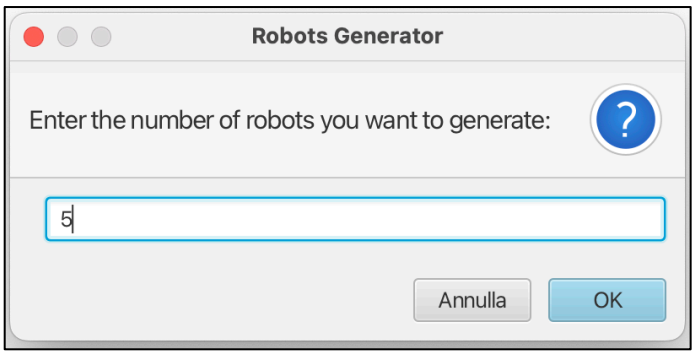

Figura 1: *Robots Generator window.*

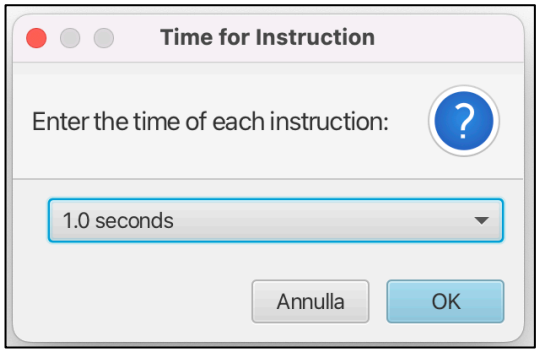

Figura 2*: Time for Instruction window.*

Una volta inseriti i dati iniziali in modo corretto uscirà fuori un'interfaccia grafica in cui sono presenti tre bottoni: il primo bottone per caricare il file dell'ambiente, il secondo bottone per caricare il programma ed eseguirlo, il terzo bottone permette di riavviare l'applicazione per poter fare una nuova simulazione.

Al termine dell'esecuzione verrà visualizzata la configurazione finale e il relativo tempo di esecuzione in secondi del programma (*Figura 3*).

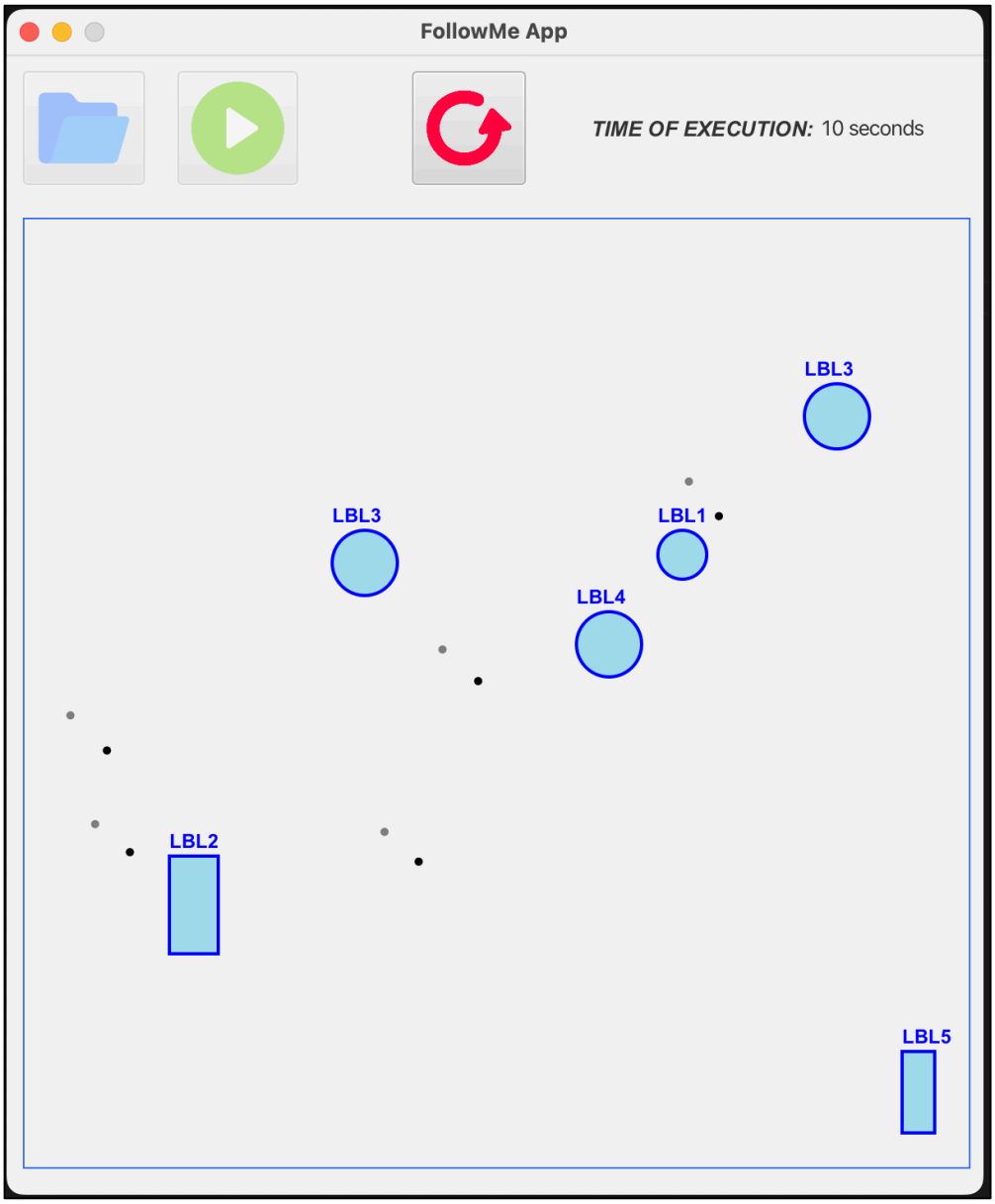

Figura 3: *FollowMe App execution finished.*

All'interno della cartella `file` sono presenti due file per poter effettuare un'esecuzione base dell'applicazione:

- `/CardelliniMatteo119137/file/environmentProgram.rshp`;
- `/CardelliniMatteo119137/file/robotProgram.rprg`.