DOCUMENTAZIONE CASO DI STUDIO
PER IL CORSO DI METODI AVANZATI
DI PROGRAMMAZIONE

DIPARTIMENTO
DI INFORMATICA

SIMONE SUMMO, MATRICOLA 716921

IL NOME

INDICE

PARTE 1 - L’AVVENTURA TESTUALE

PARTE 2 – ASPETTI TECNICI

PARTE 1 – L’AVVENTURA TESTUALE

SPUNTI E TRAMA DEL GIOCO

Questa avventura testuale prende spunto da una sezione del gioco *Paper Mario: Il portale millenario* un gioco della celebre saga che vede il personaggio di Mario come protagonista.

La storia inizia quando Mario riceve una misteriosa lettera dalla principessa Peach, che lo invita a raggiungerla nella città portuale di Fannullopoli. Peach racconta di aver scoperto una mappa del tesoro durante un viaggio e chiede a Mario di aiutarla a trovare il leggendario tesoro del Portale Millenario.

Mentre colleziona le gemme stella, pietre che servono ad attivare il Portale Millenario, Mario si scontra con una misteriosa organizzazione chiamata Società X, che ha rapito Peach per manipolarla affinché aiuti a sbloccare il portale.

Grazie alle gemme stella e ai suoi compagni di viaggio, Mario riesce a sventare i piani della Società X e a salvare Peach.

Durante il terzo capitolo della storia, Mario si reca nella città di Casalcrepuscolo che sta vivendo un periodo non facile: un mago, che si è impossessato del castello fuori città, ha lanciato una maledizione sulla città. Questa maledizione consiste nel fatto che ogni mezz’ora, al rintocco di una campana, gli abitanti della città vengono trasformati in maiali.

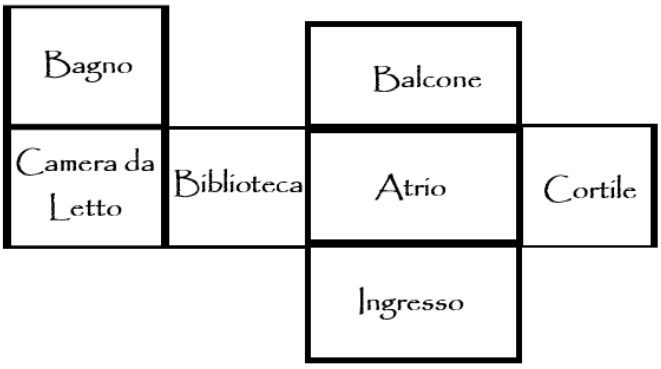
Mario arriverà al castello e incontrerà non un mago ma un fantasma, molto timido, con cui si scontrerà. Questo fantasma, che ha preso possesso anche della gemma stella, possiede un potere particolare: riesce a trasformarsi in qualsiasi essere vivente e prendere il suo posto. Quando pensa di averlo sconfitto il fantasma rivela che Mario non può sconfiggerlo e che lui non può essere sconfitto se non si pronuncia il suo nome e procede rubando a Mario il suo corpo e il suo nome e trasformandolo così in un’ombra.

Con l’aiuto di Vivian, un’altra ombra che si unirà alla sua compagnia, Mario riesce a scoprire il nome del fantasma: Rampel.

Una volta scoperto il nome, Mario va ad affrontare Rampel e sconfiggendolo riacquista il suo nome, il suo corpo e la gemma stella e così prosegue il suo viaggio alla ricerca delle rimanenti gemme stella.

Il gioco, da me creato, è una rivisitazione di questa ultima parte con lo scopo di scoprire il nome del fantasma e sconfiggerlo definitivamente, permettendogli di rientrare a casa sua.

MAPPA



COME SI GIOCA

Il giocatore potrà muoversi lungo gli ambienti della mappa con i quattro punti cardinali: nord, sud, est e ovest.
In qualsiasi stanza il giocatore avrà a disposizione una descrizione dell'ambiente attraverso il comando "guarda" che darà indicazioni sull'eventuale presenza di oggetti nella stanza e informazioni geografiche per la mossa successiva.
Ho deciso di non inserire la lista dei passi da compiere nel gioco nella documentazione per dare un'esperienza di gioco più autentica.

LISTA DEI COMANDI

Per quanto riguarda la lista dei comandi dobbiamo distinguere i comandi di movimento dai comandi d'azione.

I comandi di movimento sono i seguenti:

Comandi	Alias
Nord	nord-NORD
Sud	sud-SUD
Est	est-EST
Ovest	ovest-OVEST

Quelli di azione invece:

Comandi	Alias
Inventario	inventario-INVENTARIO-inv-INV
Guarda	guarda-GUARDA-osserva-OSSERVA-vedi-VEDI-descrivi-DESCRIVI
Raccogli	raccogli-RACCOGLI-prendi-PRENDI
Usa	usa-USA-utilizza-UTILIZZA-applica-APPLICA-inserisci-INSERISCI
Accendi	accendi-ACCENDI
Premi	premi-PREMI-spingi-SPINGI
Fine	fine-FINE-end-END-esci-ESCI-exit-EXIT

Parte 2 – Aspetti tecnici

Architettura del sistema

Questo progetto rappresenta un framework per un gioco di avventura testuale. Le classi definite all'interno del sistema lavorano insieme per creare un ambiente di gioco interattivo, dove il giocatore può esplorare stanze, raccogliere oggetti, risolvere enigmi e progredire nella trama attraverso comandi testuali. Ogni classe svolge un ruolo specifico nella struttura del gioco, permettendo un'interazione dinamica tra il giocatore e l'ambiente virtuale.

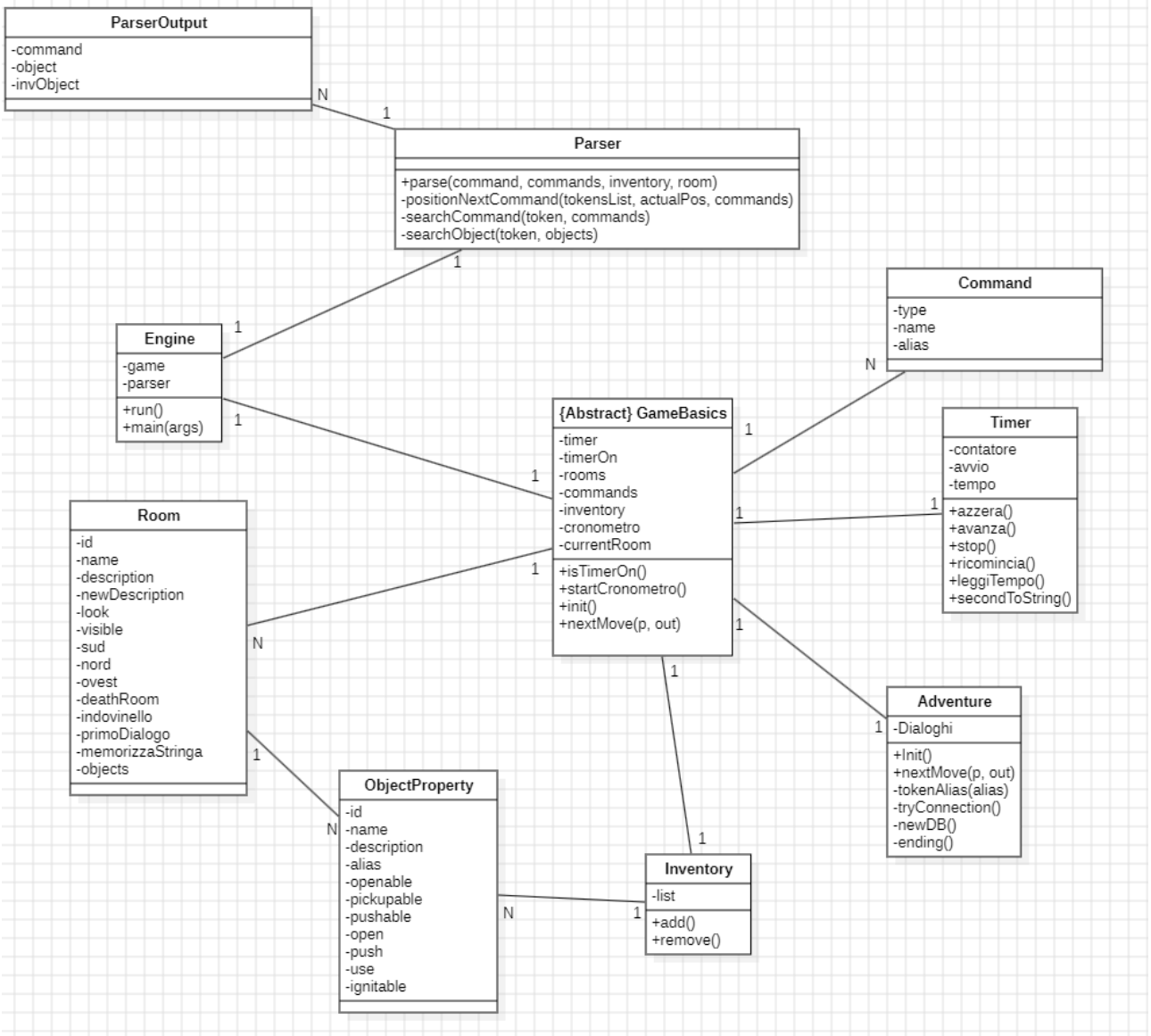
Le principali componenti del gioco sono rappresentate da classi che gestiscono vari aspetti dell'esperienza di gioco:

- Adventure: la classe principale che implementa la logica di gioco, gestendo l'inizializzazione della mappa, gli oggetti e le stanze, e le interazioni del giocatore.
- Dialoghi: responsabile della gestione dei dialoghi che vengono mostrati al giocatore durante l'avventura, adattandosi alle varie situazioni del gioco.
- Command: questa classe rappresenta i comandi che il giocatore può eseguire per interagire con il mondo di gioco.
- CommandType: definisce i tipi specifici dei comandi.
- ObjectProperty: rappresenta gli oggetti presenti nel mondo di gioco, con informazioni su proprietà quali se un oggetto è raccoglibile, utilizzabile o premibile.
- Room: gestisce le stanze del gioco. Ogni stanza ha un nome, una descrizione e può essere collegata ad altre stanze, contenere oggetti ed enigmi.
- Timer: un sistema di gestione del tempo che tiene traccia del tempo trascorso nel gioco e fornisce meccanismi per avviare, fermare o resettare il conteggio.
- Engine: il motore del gioco che avvia il ciclo di gioco, processa l'input del giocatore e chiama le classi appropriate per gestire i comandi.
- GameBasics: una classe astratta che definisce la struttura generale del gioco. Le classi che la estendono devono implementare i metodi per inizializzare il gioco e gestire le mosse del giocatore.

- Parser: interpreta i comandi inseriti dal giocatore, analizzando l'input e restituendo un output in base agli oggetti e alle stanze coinvolte.
- ParserOutput: contiene il risultato dell'interpretazione di un comando, rappresentando il comando eseguito e gli oggetti coinvolti.
- Inventory: gestisce l'inventario del giocatore, permettendo l'aggiunta, la rimozione e la consultazione degli oggetti raccolti.

Questo sistema modulare consente una facile espansione e manutenzione, rendendo possibile aggiungere nuove funzionalità o adattare il comportamento delle classi per migliorare l'esperienza di gioco.

DIAGRAMMA DELLE CLASSI



SPECIFICA ALGEBRICA

La coda è una struttura dati lineare che segue il principio FIFO, cioè il primo elemento che entra nella coda sarà anche il primo ad essere rimosso. Questo comportamento è simile ad una fila di persone in attesa, dove il primo che arriva è il primo ad essere servito. Questa struttura è stata pensata per accettare due comandi in input consecutivi dall'utente, ad esempio: "usa il martello e raccogli la torcia".

Operatori:

- Emptyqueue: crea una coda vuota
- Enqueue: aggiungi elementi ad una coda vuota
- Dequeue: elimina un elemento dalla coda
- Top: legge il valore del primo elemento nella coda
- Codavuota: operatore booleano che ci dice se la coda sia piena o vuota

Operazioni:

- Emptyqueue() -> coda
- Enqueue(coda, elemento) -> coda
- Dequeue(coda) -> coda
- Top(coda) -> elemento
- Codavuota(coda) -> boolean

Costruttori:

osservazioni	**costruttori**	
	emptyqueue()	enqueue(coda,elemento)
codavuota(coda')	true	false
top(coda')	error	**If** codavuota(coda) **then** elemento **else** top(coda)
dequeue(coda')	error	**If** codavuota(coda) **then** emptyqueue **else** enqueue(dequeue(coda),elemento)

Specifica semantica (dati una coda c, un elemento e ed un valore booleano true\false):

codavuota(emptyqueue)	true
codavuota(enqueue(c,e))	false
dequeue(enqueue(c,e))	**if** codavuota(c) **then** emptyqueue **else** enqueue(dequeue(c),e)
top(enqueue(c,e))	**if** codavuota(c) **then** e **else** top(c)

DETTAGLI DI IMPLEMENTAZIONE

Tempo di gioco

Nel contesto delle avventure testuali, è stato introdotto un nuovo componente fondamentali nel package “elements”: la classe Timer.java, che permette di implementare avventure a tempo.
Questa classe gestisce il tempo di gioco tramite metodi che sfruttano la sincronizzazione dei thread, garantendo che solo un thread per volta possa accedere a tali metodi. In questo modo, si calcola con precisione il tempo trascorso dall’inizio del gioco, fornendo al progettista uno strumento utile per impostare limiti di tempo o scadenze nelle sfide che il giocatore deve affrontare.

Tuttavia, l’uso di questa funzionalità non è obbligatorio. Se l’oggetto di tipo Timer non viene inizializzato dall’avventura, l’Engine non imporrà alcun limite di tempo durante il gioco. Questo fornisce una flessibilità maggiore in quanto si può scegliere se integrare o meno una componente temporale nella propria avventura.

Database

L’architettura dell’avventura testuale supporta aggiornamenti dinamici grazie all’uso di un database relazionale. Le descrizioni relative ai comandi, agli oggetti e alle stanze vengono caricate direttamente da un database, consentendone anche una facile manutenzione e un facile aggiornamento.

Il database utilizzato è H2, un gestore di database relazionale leggero e integrato. Questo sistema non solo offre un’archiviazione efficiente dei dati, ma mette a disposizione una console interattiva per accedere e modificare le tabelle del database in tempo reale.