Trova la Parola

# Indice

Indice 2

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 4

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Analisi e specifica dei requisiti 4

2.3 Use case 7

2.4 Pianificazione 7

2.5 Analisi dei mezzi 9

2.5.1 Software 9

2.5.2 Hardware 9

3 Progettazione 9

3.1 Design dell’architettura del sistema 9

3.2 Design dei dati e database 9

3.3 Design delle interfacce 9

3.4 Design procedurale 12

4 Implementazione 13

5 Test 13

5.1 Protocollo di test 13

5.2 Risultati test 15

5.3 Mancanze/limitazioni conosciute 15

6 Consuntivo 15

7 Conclusioni 17

7.1 Sviluppi futuri 17

7.2 Considerazioni personali 17

8 Bibliografia 18

8.1 Bibliografia per articoli di riviste: 18

8.2 Bibliografia per libri 18

8.3 Sitografia 18

9 Glossario 19

10 Indice delle figure 19

11 Allegati 19

# Introduzione

## Informazioni sul progetto

**Allievo in esame**: Nicolò Fadda

**Docente in carica**: Geo Petrini

**Scuola**: Scuola D’Arti e Mestieri Trevano

**Sezione**: Informatica

**Materia**: M306 (Progetti)

**Data di Inizio**: 01.09.2023

**Termine di Consegna**: 01.12.2023

## Abstract

* **Background**

“Trova La Parola” is an application designed to assist children in their daily activities, both curricular and extracurricular, through an educational game. The project’s goal is to use the game as a tool to improve young people’s daily lives by offering them a fun and engaging learning experience.

**Description of the problem and motivation**

The core of the project lies in the challenge of creating a game that is both fun and educational. Its importance is based on the ability to positively influence children’s lives by improving their learning skills. The main difficulty lies in balancing the playful aspect with the educational objective. The project’s success could have a significant impact on young people’s lives and their ability to learn.

**Approach/Methods**

To progress, approaches and methods aimed at combining the fun aspect of the game with its educational purpose have been employed. The work involved designing an intuitive interface, selecting educational words, and creating levels suitable for users’ ages. The process required careful evaluation of what to control, ignore, or measure, taking into account the crucial factors for the project’s success.

**Results**

The result is an application that offers an effective balance between entertainment and learning. “Trova La Parola” is proposed as an option that improves speed, security, and overall experience compared to other similar solutions. Its implementation provides an answer to children’s learning needs, offering a product that represents a significant step forward compared to existing solutions.

## Scopo

Lo scopo didattico del progetto è riuscire a gestire nel modo più ottimale possibile un progetto IT con le risorse datoci dalla scuola.

Questo servirà successivamente alla preparazione in vista dell’esame pratico al quarto anno scolastico.

Lo scopo operativo riguarda il creare un applicativo dove si possa stampare il gioco “Trova la parola” classico ma in formato digitale (e cartaceo).

# Analisi

## Analisi del dominio

L’applicativo potrà essere utilizzato da chiunque voglia provare il gioco delle parole crociate.

Attualmente esistono prodotti simili ma sono principalmente web e con una necessità di avere una connessione ad internet perciò questo applicativo sarà offline e stampabile.

## Analisi e specifica dei requisiti

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-01 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Creare l’interfaccia | | | | |
| **Note** | Riquadro attorno alla griglia e aggiunta di multi-colonne per le parole | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Si necessita di controllare la grandezza | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-02 | **Priorità** | 3 | **Versione** | 1.0 |
| **Nome** | Creare la matrice di parole | | | | |
| **Note** | Random; Utente può passare delle parole a sua scelta | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Si necessita che la grandezza sia equivalente ad un foglio A4 | | | | |
| **002** | Si necessita che ci siano lettere tutte in maiuscolo | | | | |
| **003** | Si necessita che non ci sia nessun accento o apostrofi | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-03 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Creare livelli in base alla difficoltà | | | | |
| **Note** | Difficoltà bambini e normale | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Mod. bambini non ha una parola finale; Mod. normale l’avrà | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-04 | **Priorità** | 3 | **Versione** | 1.0 |
| **Nome** | Rendere l’applicativo multipiattaforma | | | | |
| **Note** | Usare linguaggi che lo permettono come Java o Python | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-05 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Rendere le parole da trovare in ordine alfabetico | | | | |
| **Note** | - | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-06 | **Priorità** | 3 | **Versione** | 1.0 |
| **Nome** | Parole trovabili in ogni direzione | | | | |
| **Note** | Incroci nella stessa direzione non possibile | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-07 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Scelta del font | | | | |
| **Note** | Monospace di default ma l’utente può cambiarlo | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-08 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Export | | | | |
| **Note** | Deve essere .txt o .png | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-09 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Dimensione della griglia | | | | |
| **Note** | Può essere a discapito dell’utente | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Fare i controlli di dimensione per un foglio A4 | | | | |
| **002** | È a dipendenza della difficoltà | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-10 | **Priorità** | 3 | **Versione** | 1.0 |
| **Nome** | Margine di errore | | | | |
| **Note** | Non deve generare errori di compilazione/semantici/… | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-11 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Dizionario parole | | | | |
| **Note** | Può essere usato un file da cui prendere le parole; | | | | |

**Spiegazione elementi tabella dei requisiti:**

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

## Use case

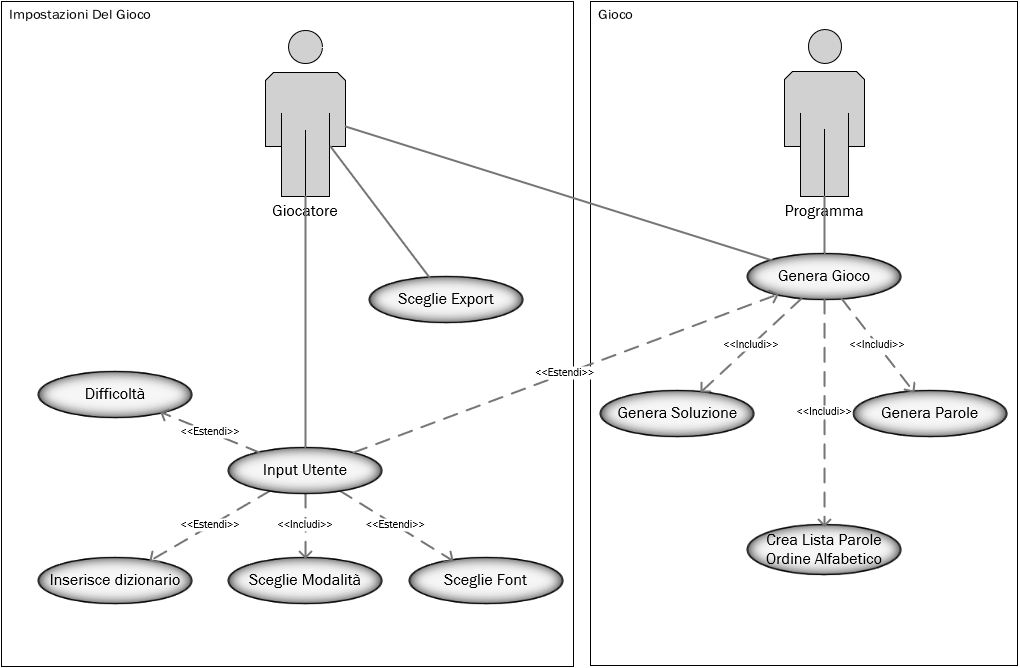


Figura 1 - Use Case

## Pianificazione

(Vedi pagina seguente)

Gannt Preventivo

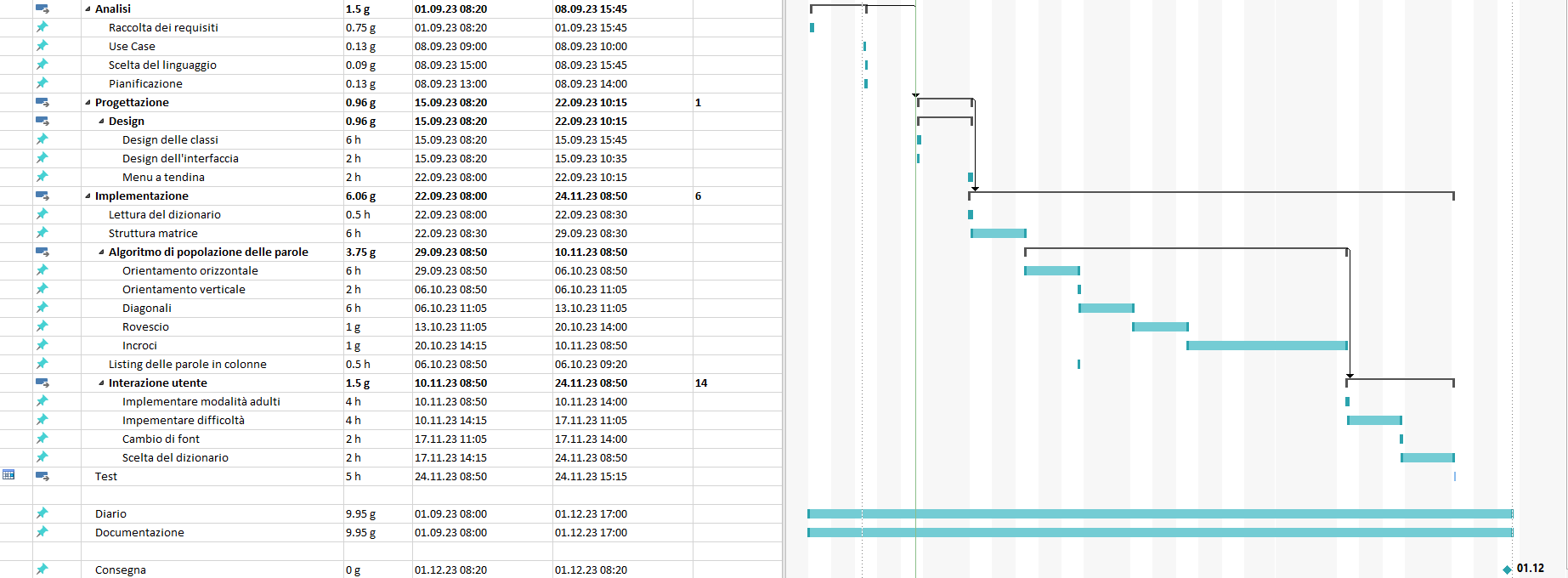


Figura 2 - Gannt Preventivo

## Analisi dei mezzi

Per la realizzazione del progetto "Trova La Parola", sono stati impiegati diversi mezzi, tra software e hardware, per garantire un ambiente di sviluppo adeguato e efficiente.

È stata utilizzata la metodologia waterfall per lo sviluppo dei gannt.

### Software

JDK19

Java versione 11.0.16.1

Netbeans (IDE)

GitHub

### Hardware

Un pc con le seguenti specifiche:

* Processore Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz 3.00 GHz
* 32GB RAM
* Sistema operativo a 64 bit

# Progettazione

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all’esecutore di evitare fraintendimenti e imprecisioni nell’implementazione del prodotto.

## Design dell’architettura del sistema

-

## Design dei dati e database

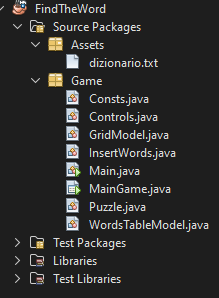


Figura 3 - Organizzazione dei dati

Sono stati utilizzati vari package come “Assets” per il dizionario e “Game” come package principale dove sono contenute tutte le classi principali.

## Design delle interfacce

Per questo progetto ho realizzato una sola interfaccia, ovvero l’interfaccia finale che vedrà l’utente una volta avviato il programma.

UML iniziale:

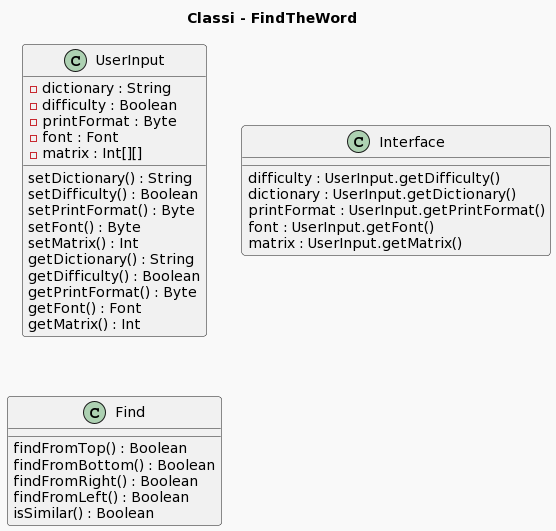


Figura 4 - Design delle interfacce

UML finale (vedi pagina successiva):

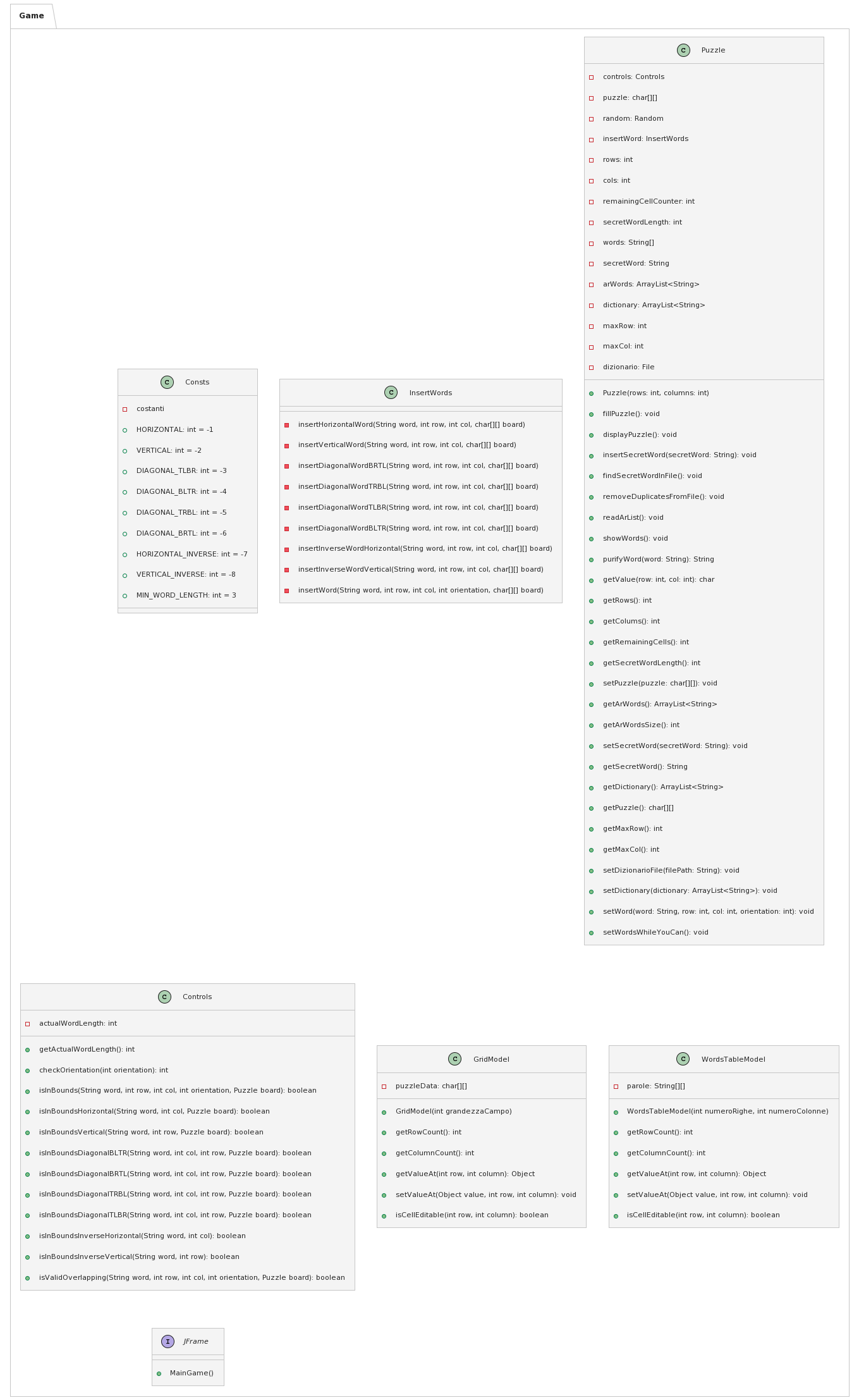


Figura 5 - Design delle interfacce - finale

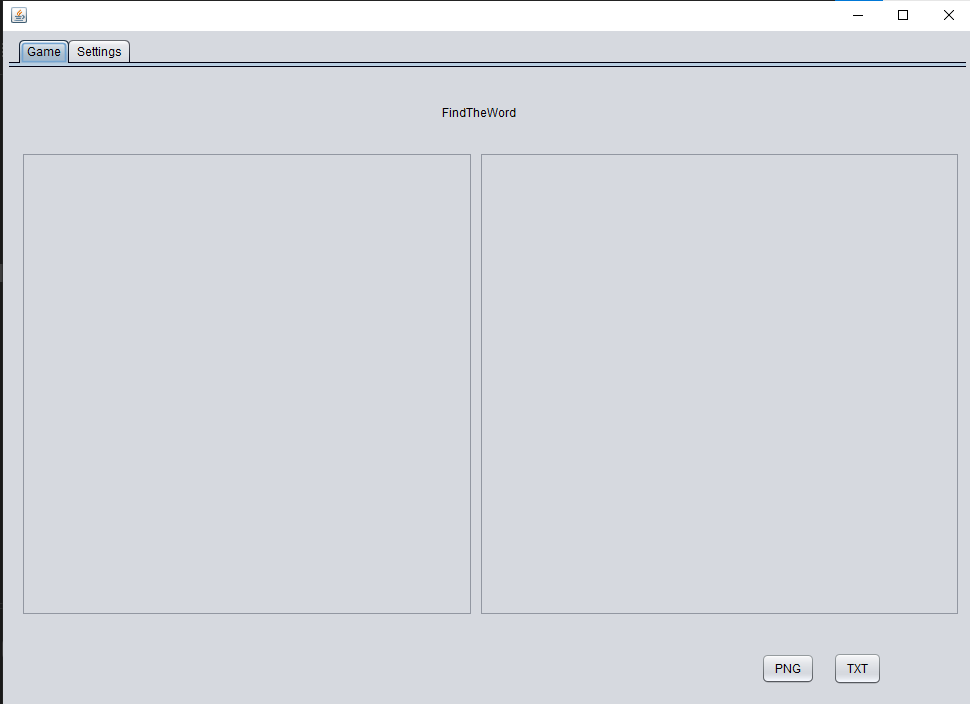


Figura 6 - Interfccia all'avvio

Interfaccia principale al primo avvio (Game)

Questa interfaccia è l’interfaccia principale dove, quando viene cliccato il bottone “Genera” nell’interfaccia “Settings”, verrà generato il codice.

Nella tabella di sinistra sarà presente la griglia di parole, ovvero la parte saliente del gioco.

Nella tabella di destra invece saranno presenti le parole da trovare nella griglia di sinistra inserite in forma di colonna e in ordine alfabetico come richiesto dai requisiti.

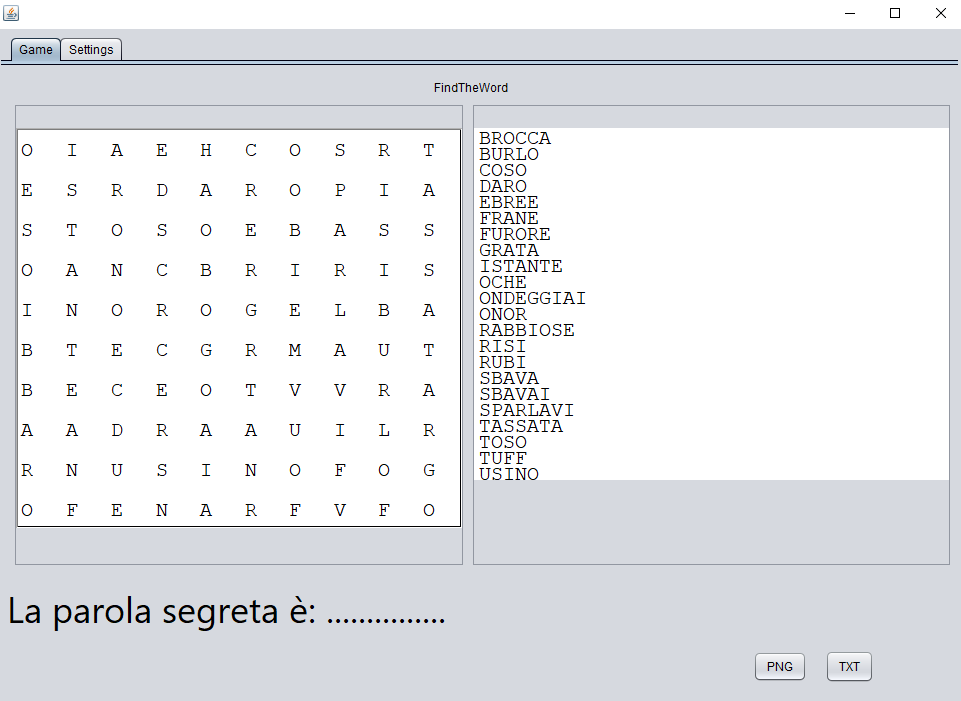


Figura 7 - Interfaccia generata

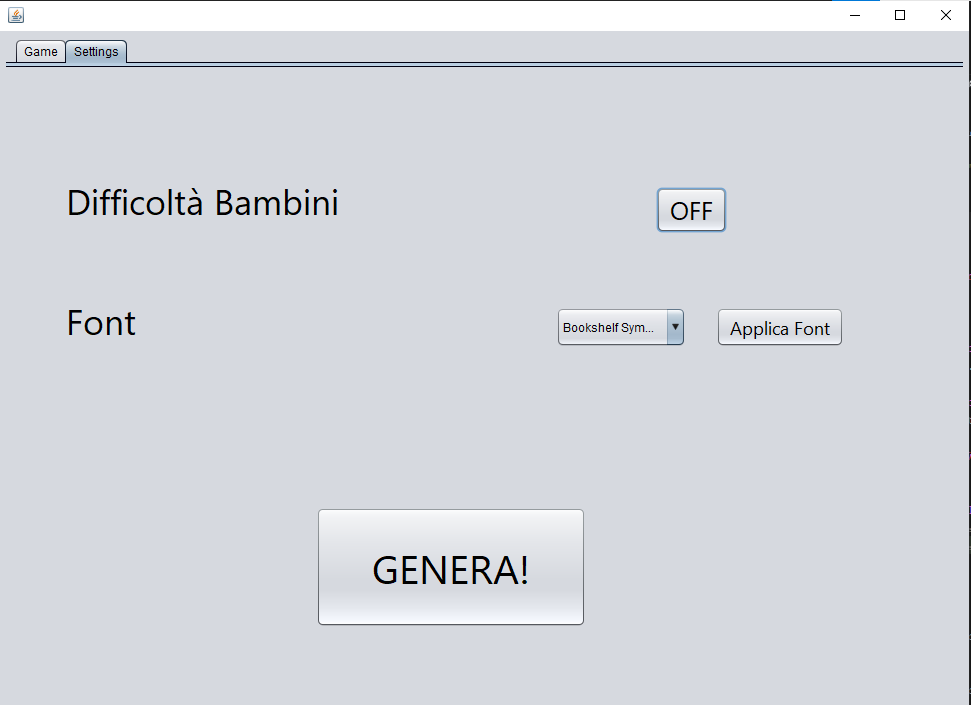


Figura 8 - Interfaccia settings

Interfaccia di impostazioni (Settings)

Questa interfaccia viene utilizzata per avere le varie impostazioni del gioco e per generare il gioco in se’.

La difficoltà bambini rappresenta un bottone (ON e OFF) dove permetterà la scelta dell’inserimento di una parola segreta all’interno del gioco.

**ON 🡪 Parola segreta attiva**

**OFF 🡪 Parola segreta disattivata**

In seguito c’è la sezione dedicata al Font dei caratteri del gioco.

Questo, tramite un Combo Box può essere scelto e applicato al gioco liberamente tramite un set di Font di default della macchina dell’utente.

In fine è presente il bottone “Genera”, che come dice il nome, serve a generare il gioco nella tab “Game”.

Esso è l’evento principale del gioco, siccome crea e popola le tabelle nella sezione “Game”.

## Design procedurale

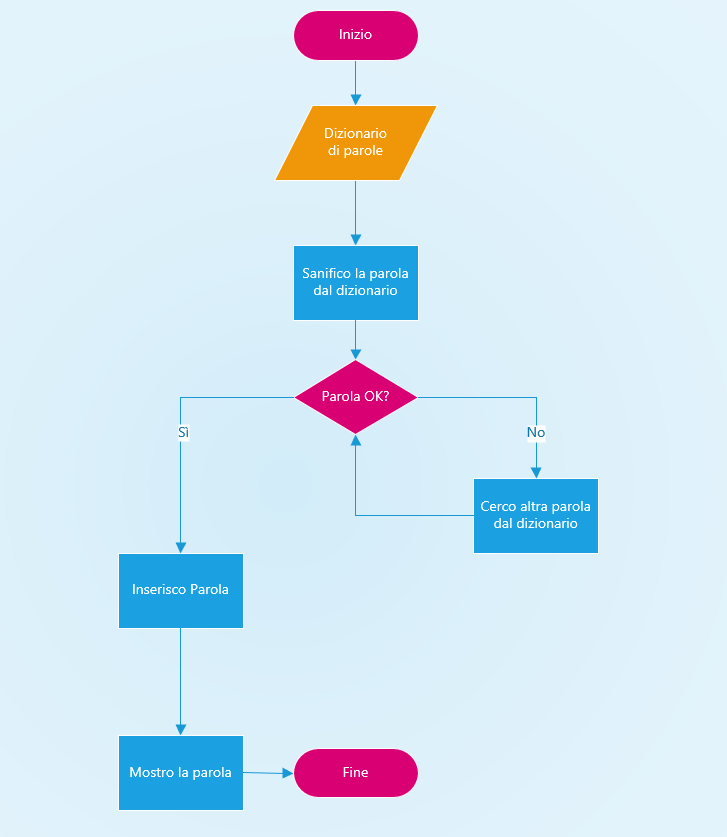


Figura 9 - Design procedurale

Procedura principale

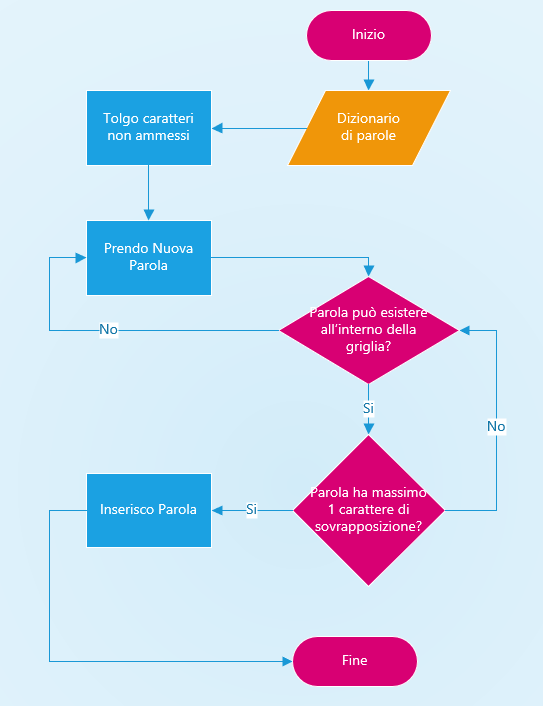


Figura 10 - Design algoritmo

Procedura interna (Algoritmo)

# Implementazione

## Classe Consts

### Metodi

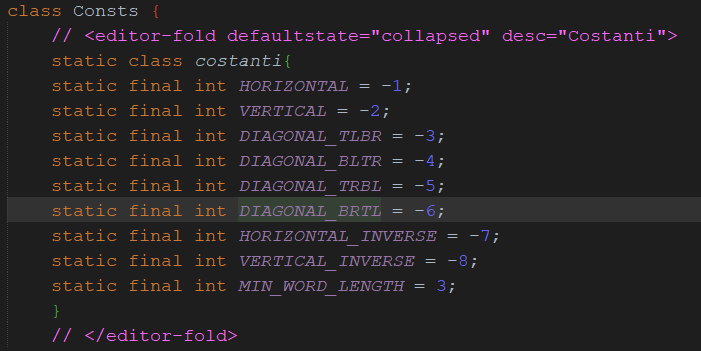


Figura 11 - Classe Consts

Questa classe è importante per la pulizia di codice e funzionamento del programma: essa contiene tutte le costanti utili.

Queste costanti sono costantemente utilizzate nel codice soprattutto per controllare le varie direzioni delle parole e controllare la griglia principale.

## Classe Controls

### Metodi

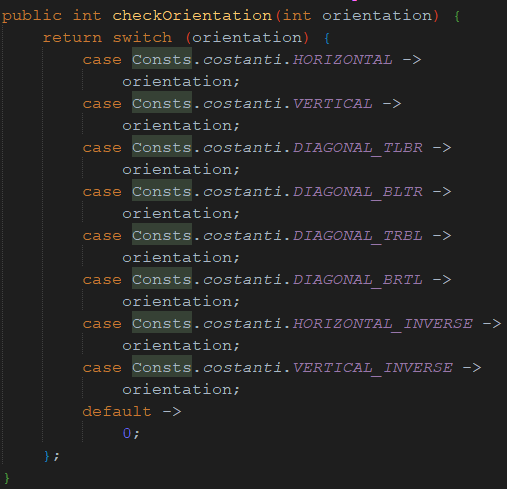


Figura 12 - Metodo CheckOrientation

Questo metodo controlla semplicemente se il numero passato al metodo esiste e ha una funzione specifica: metodo di controllo.

Ritorna lo stesso numero che sarebbe l’orientamento.

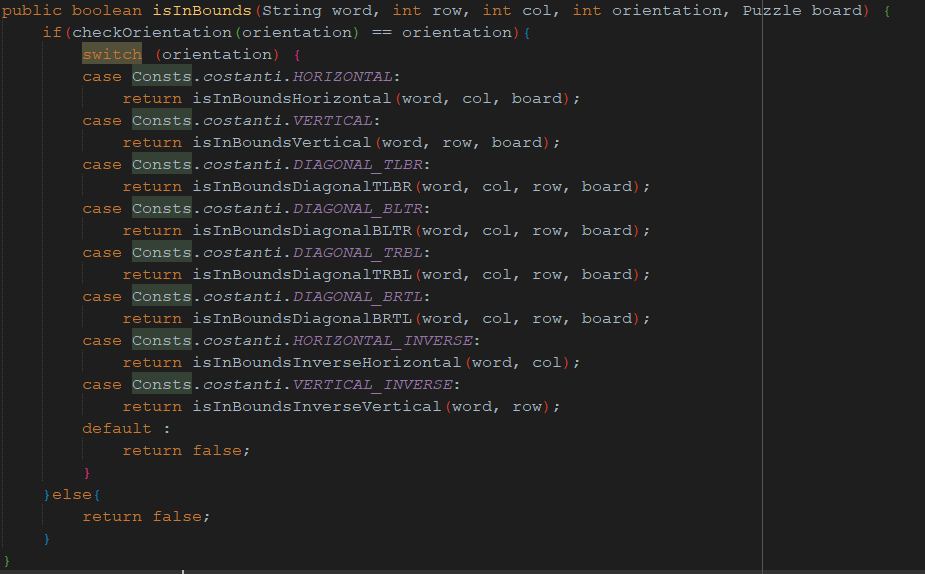


Figura 13 - Metodo isInBounds

Questo metodo controlla se la parola passata può esistere all’interno della griglia finale (come dice il nome “isInBounds”) utilizzando dei sotto-metodi per controllare ogni direzione della griglia.

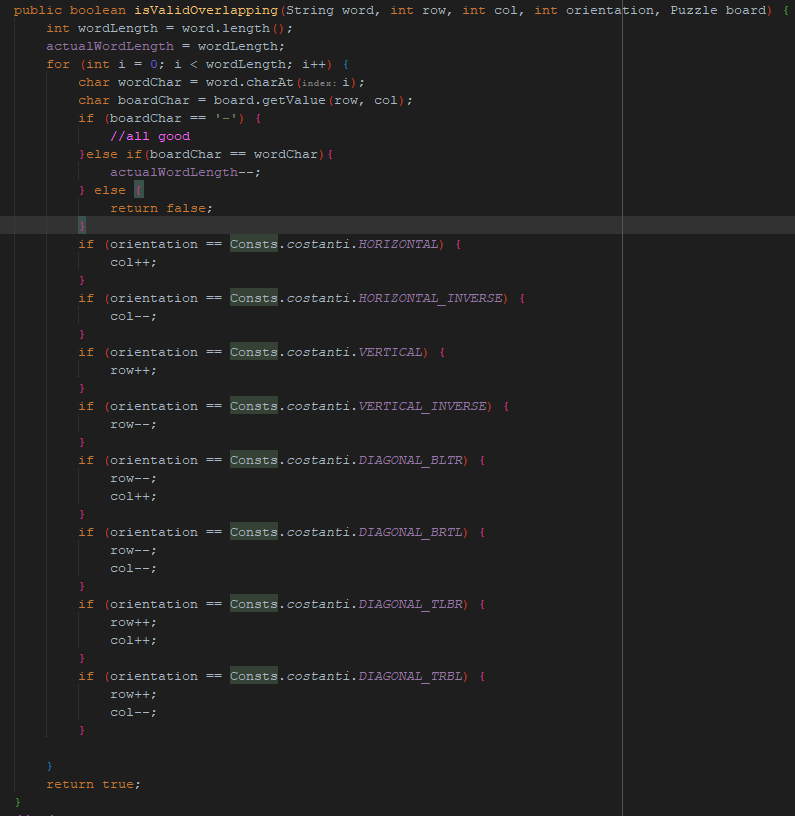


Figura 14 - isValidOverlapping

Questo metodo controlla se la parola da inserire potrebbe sovrapporsi con una parola già presente nella griglia, controllando cella per cella a dipendenza dell’orientamento della parola.

In caso sia valida la sovrapposizione, ritorna true.

## Classe InsertWords

### Metodi

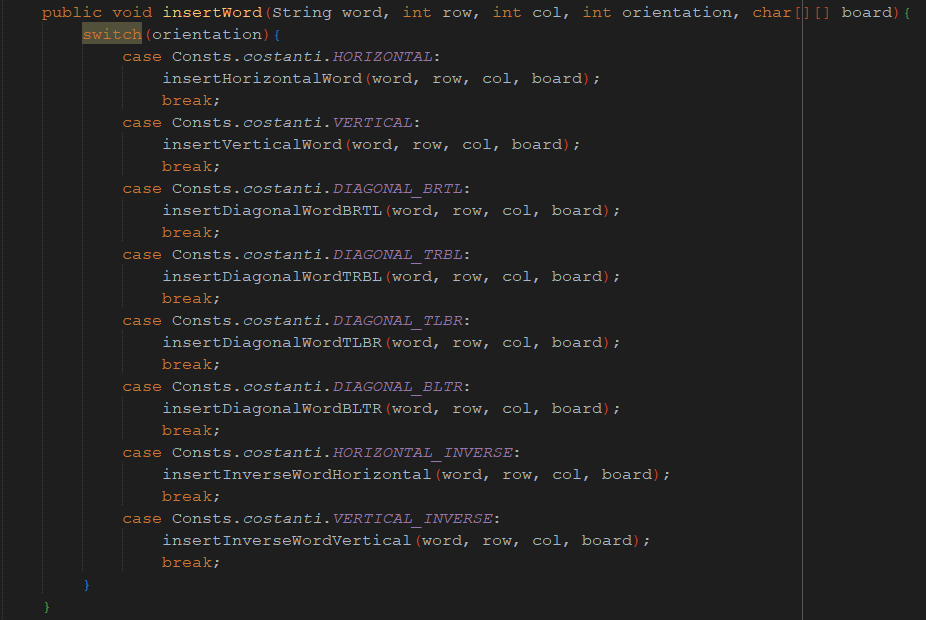


Figura 15 - Metodo insertWord

Questo è il metodo principale per inserire le parole nella griglia.

Esso a dipendenza dell’orientamento della parola utilizza il metodo singolo per l’inserimento della parola in quella determinata direzione.

## Classe Puzzle

### Metodi

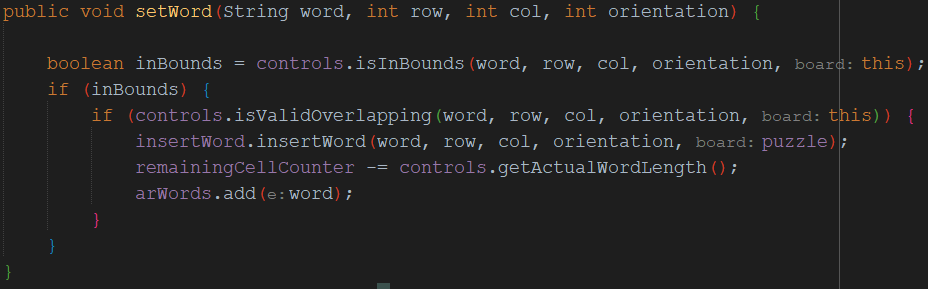


Figura 16 - Metodo setWord

Il seguente metodo serve a fare tutti i controlli necessari alla parola che dovrà essere inserita poi nella griglia.

Il metodo inserisce la parola singola all’interno della griglia.

Verrà usato per il metodo seguente:

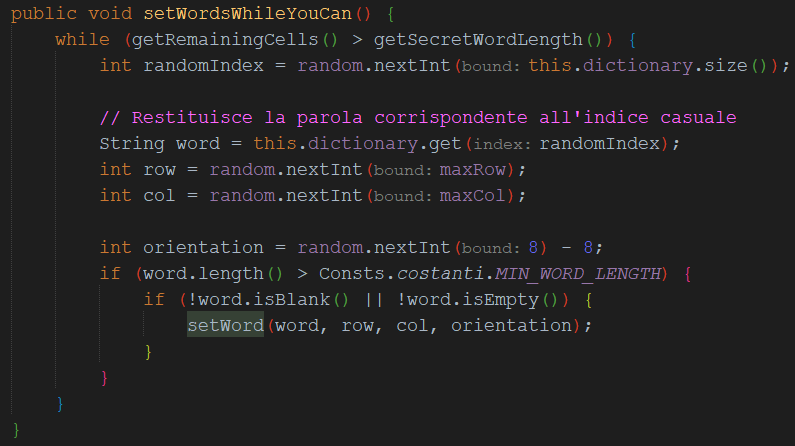


Figura 17 - Metodo setWordsWhileYouCan

Questo è il metodo principale del funzionamento del programma.

Esso è il metodo più importante di tutto il programma.

Il metodo continua a inserire le parole all’interno della griglia affinchè ci siano abbastanza celle per fare in modo che si possa inserire una parola segreta.

Perciò questo metodo utilizza il metodo precedente (setWord) per inserire, come dice il metodo “WhileYouCan” le parole.

## Classe GridModel

****

Figura 18 - Classe GridModel

Questa classe serve come modello per la tabella dell’interfaccia a sinistra dove verrà inserita la griglia del gioco.

Implementa il metodo “setValueAt” che serve principalmente a inserire un valore in una determinata riga-colonna.

## Classe WordsTableModel



Figura 19 - Classe WordsTableModel

Simile alla classe precedente ma utilizzata dalla tabella di destra, dove vengono mostrate le parole da trovare del gioco.

Esso implementa anche il metodo setValueAt.

## Interfaccia MainGame

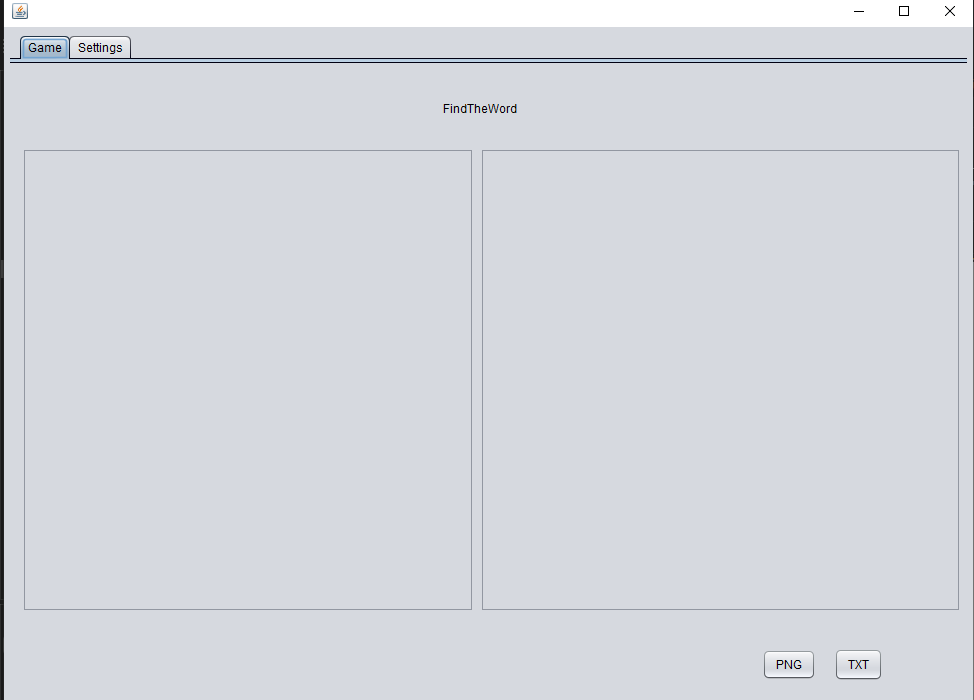


Figura 20 - Interfaccia MainGame

Questa è l’interfaccia principale che si vedrà al primo avvio (Game).

Contiene i pulsanti di export in formato PNG e TXT.

In alto a sinistra si può scegliere tra le due schermate di gioco e impostazioni.

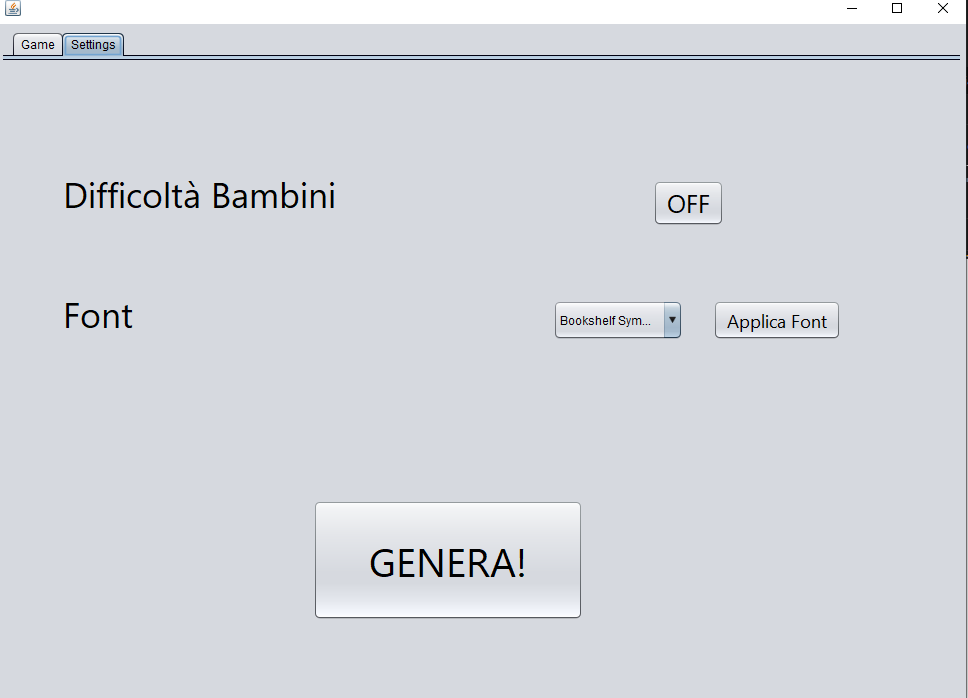


Figura 21 - Interfaccia Settings

Questa è l’interfaccia di settings, dove vengono impostate le varie impostazioni.

Contiene un bottone ON e OFF per la difficoltà e la scelta del font.

Inoltre esiste il bottone “Genera” che sarebbe il cuore del programma, esso genera il gioco nella tabella “Game” una volta premuto.

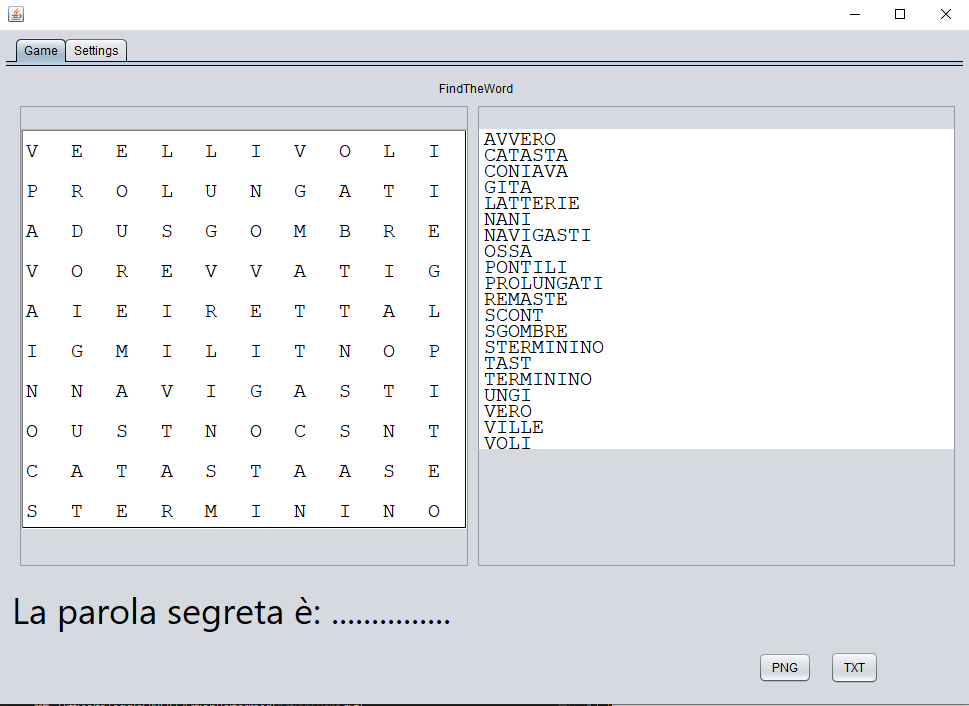


Figura 22 - Interfaccia Game Avviata

### Metodi

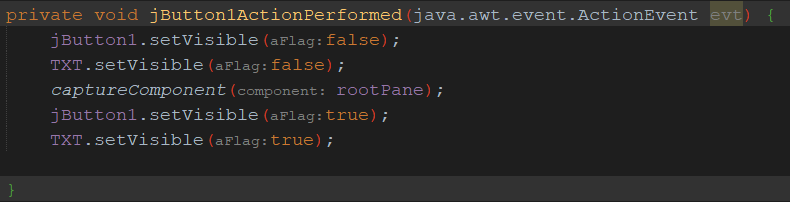


Figura 23 - Evento Bottone PNG

Questo evento tramite il metodo “captureComponent” cattura l’immagine del gioco per poi esportarla in PNG.

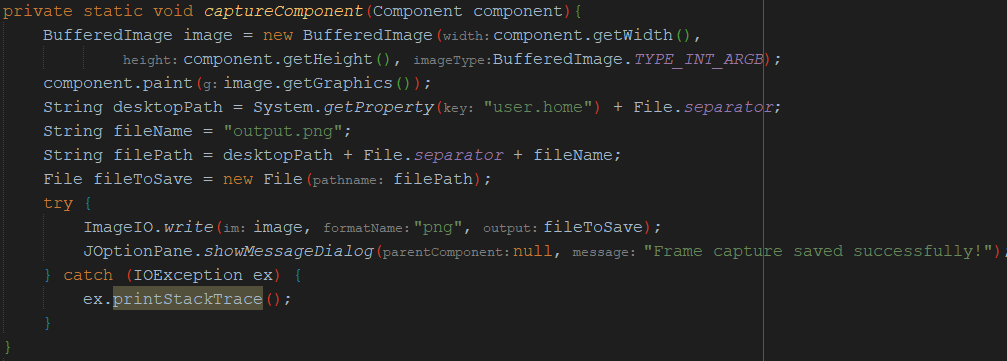


Figura 24 - Metodo captureComponent

Il metodo captureComponent crea un’immagine della schermata del componente che gli viene passato (nel mio caso rootPane che si riferisce al pannello principale del jFrame) e viene esportato in PNG nei download del profilo attuale.

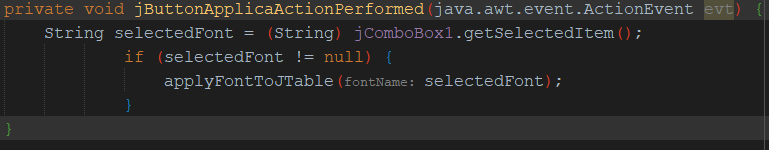


Figura 25 - Evento Selezione Font

Questo evento serve a fare in modo che il font selezionato venga applicato a tutte le tabelle nel pannello “Game” tramite il metodo “applyFontToJTable”.

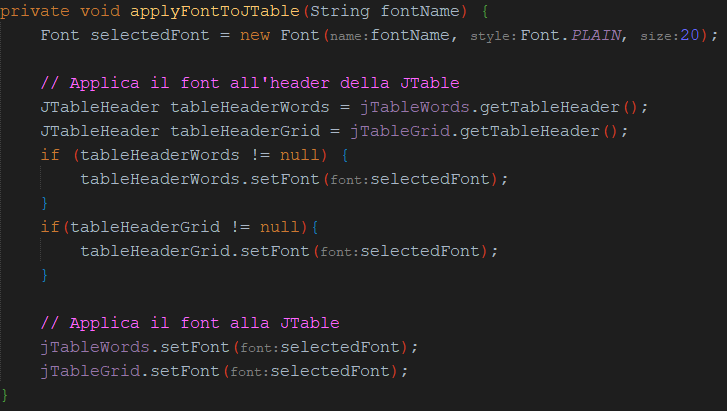


Figura 26 - Metodo applyFontToJTable

Questo è il metodo che applica i font alle tabelle.

Viene salvato nella variabile “selectedFont” il font selezionato per poi applicare il font alle varie tabelle.

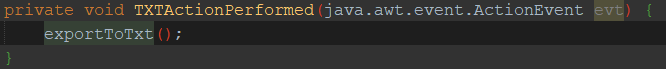


Figura 27 - Evento TXTActionPerformed

Questo è l’evento che richiama il metodo “exportToTxt” che come dice il nome, una volta cliccato il bottone esporta in txt il gioco.

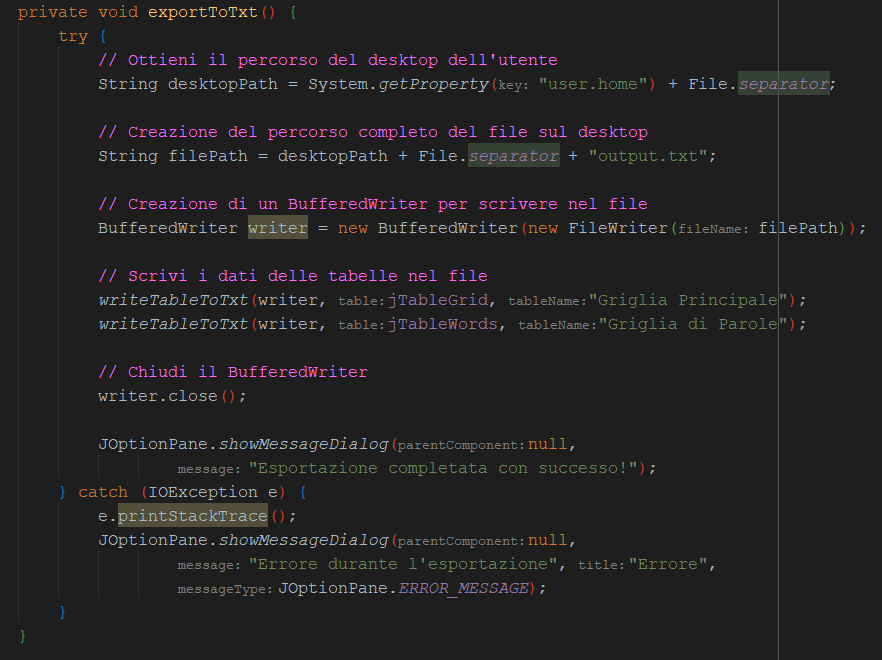


Figura 28 - Metodo exportToTxt

Il metodo utilizza un altro metodo chiamato “writeTableToTxt” per esportare tramite un percorso designato i dati delle tabelle nel file.

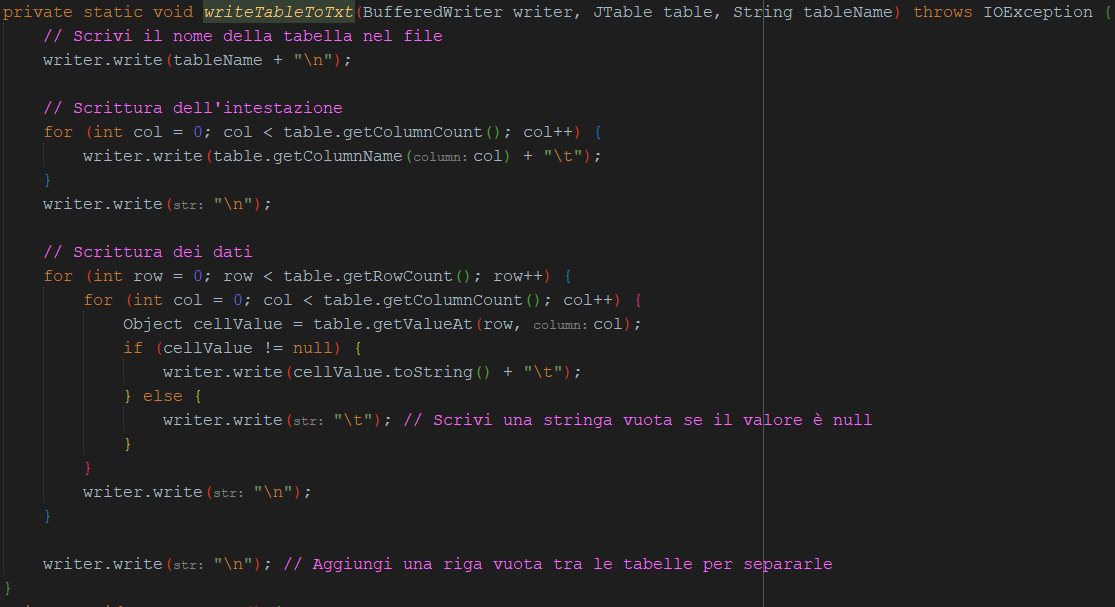


Figura 29 - Metodo writeTableToTxt

Il metodo scrive il nome della tabella nel file, seguito dall’intestazione della tabella. Successivamente, scrive i dati della tabella.

Se il valore di una cella è null, scrive una stringa vuota al suo posto.

Infine, aggiunge una riga vuota per separare le tabelle nel file.

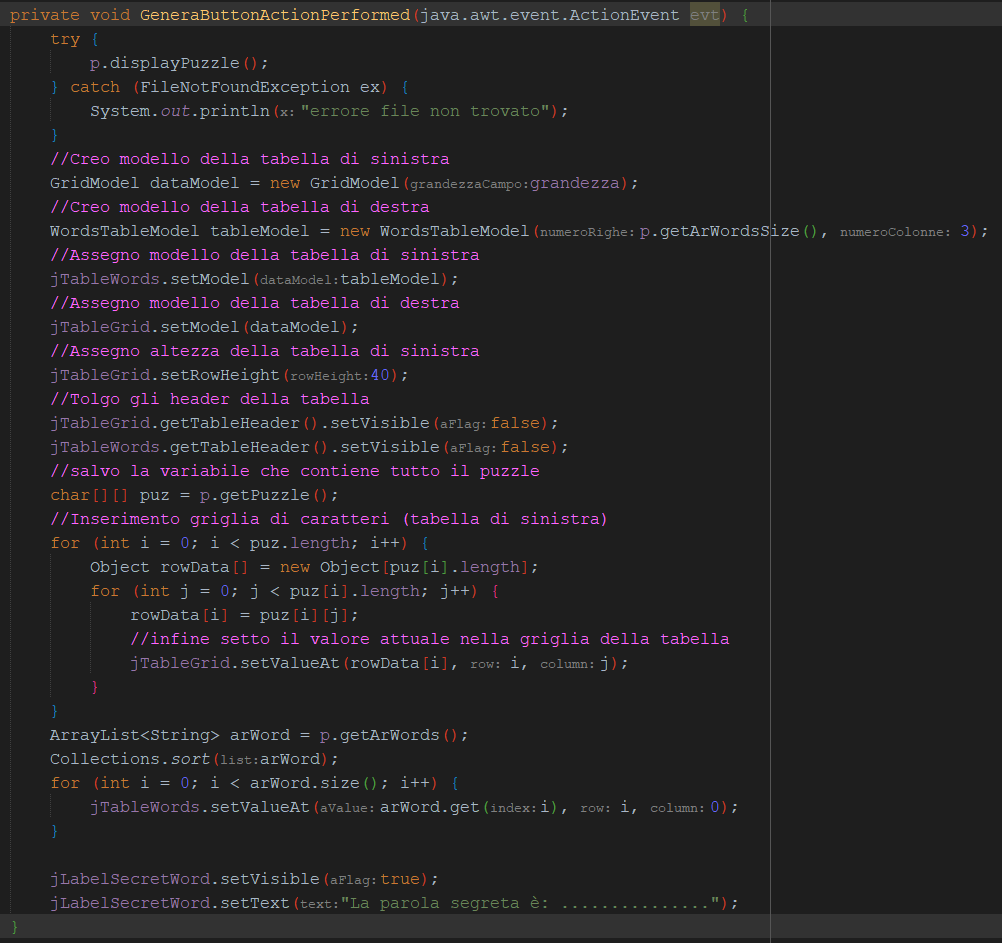


Figura 30 - Evento Bottone Genera

Questo evento è un metodo che viene chiamato quando l’utente preme il pulsante “GeneraButton”.

In questo metodo, viene creato un modello di tabella per la tabella di sinistra e un modello di tabella per la tabella di destra.

Successivamente, questi modelli vengono assegnati alle rispettive tabelle.

L’altezza della tabella di sinistra viene impostata su 40 pixel e gli header delle tabelle vengono nascosti.

Infine, il metodo popola le tabelle con i dati del puzzle e delle parole.

Viene anche impostata una JLabel per visualizzare la parola segreta.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-001  REQ-01 | **Nome** | Avvio del programma |
| **Descrizione** | Avviare il programma | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” | | |
| **Risultati attesi** | Il programma dovrebbe avviarsi senza problemi | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-002  REQ-02 | **Nome** | Generazione griglia e parole |
| **Descrizione** | Al click del bottone di generazione le parole e la griglia devono generarsi | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe generare le parole senza problemi | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-003  REQ-03 | **Nome** | Difficoltà bambini |
| **Descrizione** | Al click del bottone ON e OFF della modalità bambini deve attivarsi e disattivarsi la parola segreta | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” 5. Cliccare il bottone della modalità bambini “ON/OFF” | | |
| **Risultati attesi** | Il programma dovrebbe mostrare e nascondere il jLabel con la richiesta della parola segreta | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-004  REQ-07 | **Nome** | Font |
| **Descrizione** | Al cambiamento del font, tutte le parole nelle tabelle devono cambiare Font | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” 5. Scegliere I vari font dal Combo Box 6. Applicare il font | | |
| **Risultati attesi** | Il programma dovrebbe cambiare font correttamente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-005  REQ-05 | **Nome** | Parole in ordine alfabetico |
| **Descrizione** | Alla generazione del gioco, le parole listate devono essere in ordine alfabetico | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe generare le parole in ordine alfabetico | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-006  REQ-06 | **Nome** | Direzioni delle parole |
| **Descrizione** | Alla generazione del gioco, le parole da trovare devono essere possibili in ogni direzione | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe generare le parole in ogni direzione possibile | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-007  REQ-08 | **Nome** | Export |
| **Descrizione** | Deve essere ossible fare un export PNG | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” 5. Cliccare il bottone “PNG” | | |
| **Risultati attesi** | Il programma dovrebbe fare l’export in formato PNG | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-008  REQ-09 | **Nome** | Export in TXT |
| **Descrizione** | Deve essere ossible fare un export TXT | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” 5. Cliccare il bottone “TXT” | | |
| **Risultati attesi** | Il programma dovrebbe fare l’export in formato TXT | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-009  REQ-10 | **Nome** | Avvio senza errori di compilazione |
| **Descrizione** | Il programma deve avviarsi senza errori di compilazione | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19 | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe creare il gioco senza problemi | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-010  REQ-11 | **Nome** | Uso del dizionario |
| **Descrizione** | Il programma deve permettere all’utente di usare un dizionario personale | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, Dizionario personale di parole | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Scegliere il dizionario personale 5. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe creare il gioco con il dizionario di parole personale | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-011  REQ-11 | **Nome** | Formato file dizionario personale |
| **Descrizione** | Il programma deve permettere solo l’utilizzo di file txt come dizionario | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, Dizionario personale di parole in txt | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe permettere solo l’uso di file txt come dizionario | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-012  REQ-09 | **Nome** | Scelta grandezza griglia |
| **Descrizione** | Il programma deve permettere la scelta della grandezza della griglia | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe permettere la scelta della grandezza della griglia | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-013  REQ-09 | **Nome** | Dimensione compatibile con foglio A4 |
| **Descrizione** | Il programma deve avere una dimensione simile ad un foglio A4 | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe avere una dimensione simile ad un foglio A4 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-014  REQ-01 | **Nome** | Multicolonne |
| **Descrizione** | Il programma deve avere delle multicolonne per le parole listate | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe avere più colonne con le parole suddivise sulle varie colonne | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-015  REQ-002 | **Nome** | Caratteri in maiuscolo |
| **Descrizione** | Il programma deve trasformare tutte le lettere in maiuscolo | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe trasformare tutte le lettere in maiuscolo all’avvio del programma | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-016  REQ-003 | **Nome** | Caratteri in maiuscolo |
| **Descrizione** | Il programma deve trasformare tutti I caratteri speciali in caratteri utilizzabili e togliere gli accenti | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe trasformare tutti I caratteri speciali in caratteri utilizzabili e togliere gli accenti | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-017  REQ-06 | **Nome** | Incroci |
| **Descrizione** | Il programma non deve creare incroci nella stessa direzione e sovrapporre più di una lettera la quale deve essere la stessa | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma dovrebbe sovrapporre solo una lettera ed essa deve essere la stessa | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-018  REQ-003 | **Nome** | Font Monospace |
| **Descrizione** | Il programma deve permettere solo font monospace | | |
| **Prerequisiti** | IDE Netbeans, JAVA JDK19, | | |
| **Procedura** | 1. Avviare NETBEANS 2. Importare il programma 3. Avviare la classe “MainGame” 4. Scegliere un font 5. Cliccare il bottone “Genera” | | |
| **Risultati attesi** | Il programma genera correttamente le parole con I font solo monospace | | |

## Risultati test

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 | Il programma si è avviato senza problemi | Passato |
| TC-002 | Il programma genera correttamente griglia e parole | Passato |
| TC-003 | Il programma mostra e nasconde correttamente il jLabel | Passato |
| TC-004 | Il programma cambia font correttamente | Passato |
| TC-005 | Il programma genera correttamente le parole in ordine alfabetico | Passato |
| TC-006 | Il programma genera correttamente le parole nelle varie direzioni | Passato |
| TC-007 | Il programma genera correttamente l’export in PNG | Passato |
| TC-008 | Il programma genera correttamente l’export in TXT | Passato |
| TC-009 | Il programma si avvia correttamente senza errori di compilazione | Passato |
| TC-010 | Il programma non permette la scelta del dizionario personale | Non Passato |
| TC-011 | Il programma permette correttamente solo l’uso di file “.txt” | Passato |
| TC-012 | Il programma non permette la scelta della grandezza della griglia | Non Passato |
| TC-013 | Il programma ha una dimensione simile ad un foglio A4 | Passato |
| TC-014 | Il programma ha delle multi colonne ma ne viene usata solo una | Parzialmente Passato |
| TC-015 | Il programma trasforma correttamente tutte le lettere in maiuscolo | Passato |
| TC-016 | Il programma trasforma correttamente tutti caratteri speciali in caratteri normali / utilizzabili | Passato |
| TC-017 | Il programma non crea incroci nella stessa direzione | Passato |
| TC-18 | Il programma usa correttamente solo font monospace | Passato |

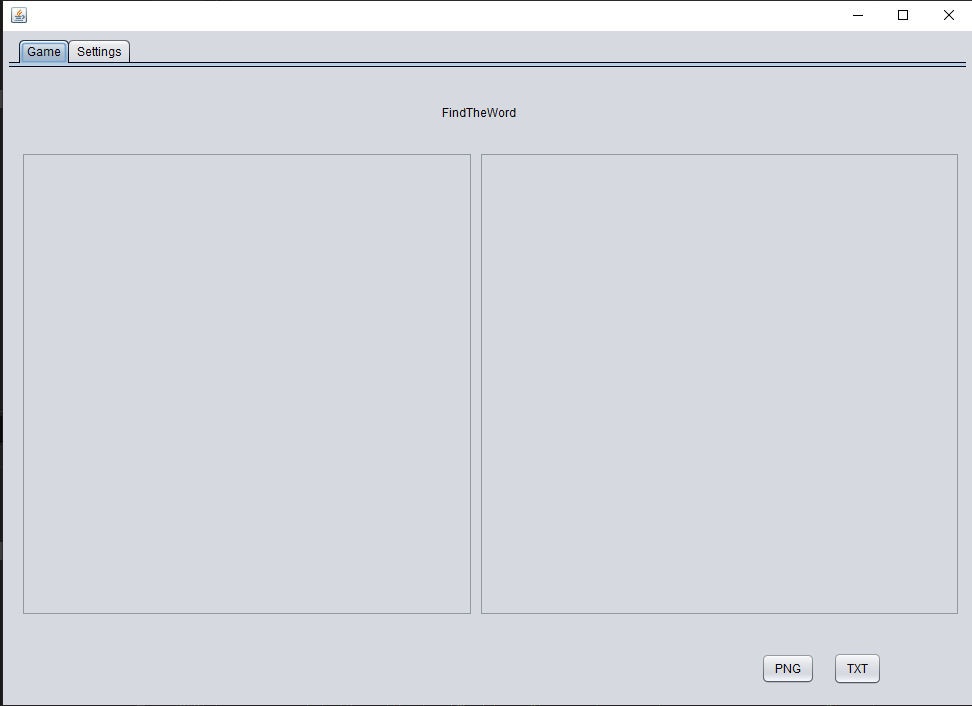


Figura 31 - TC-01

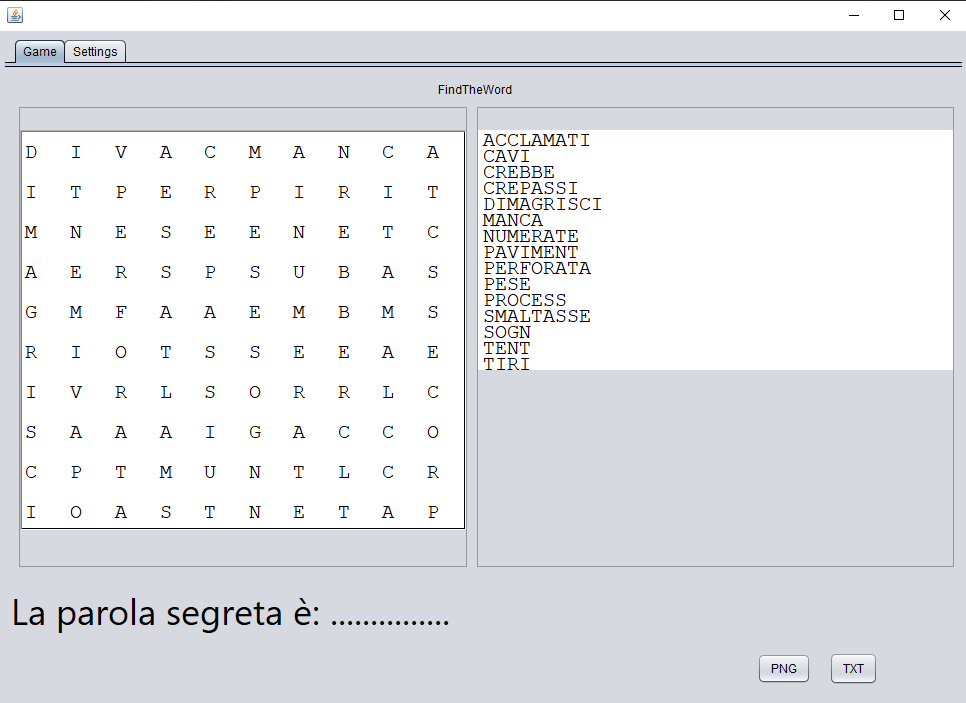


Figura 32 - TC-02

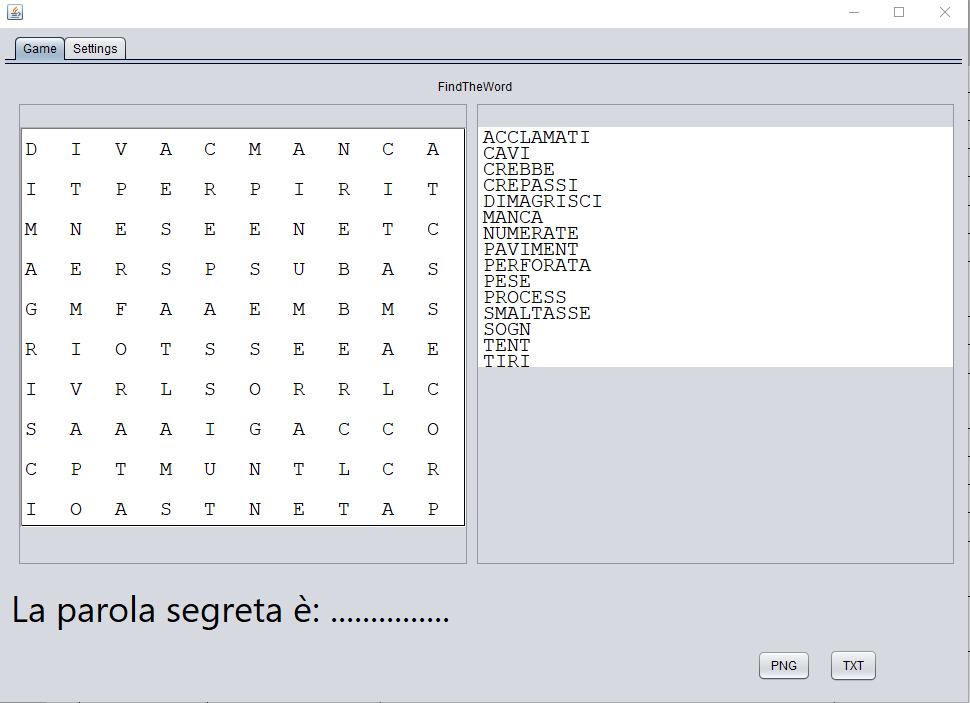


Figura 33 - TC-03

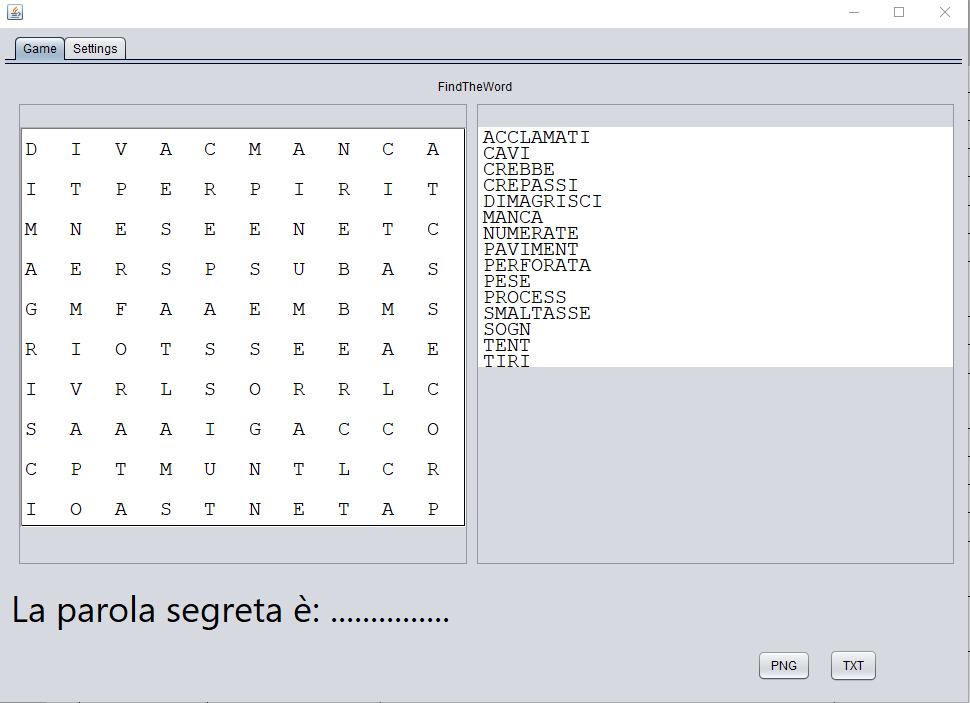


Figura 34 - TC-04 – Prima



Figura 35 - TC-04 – Dopo

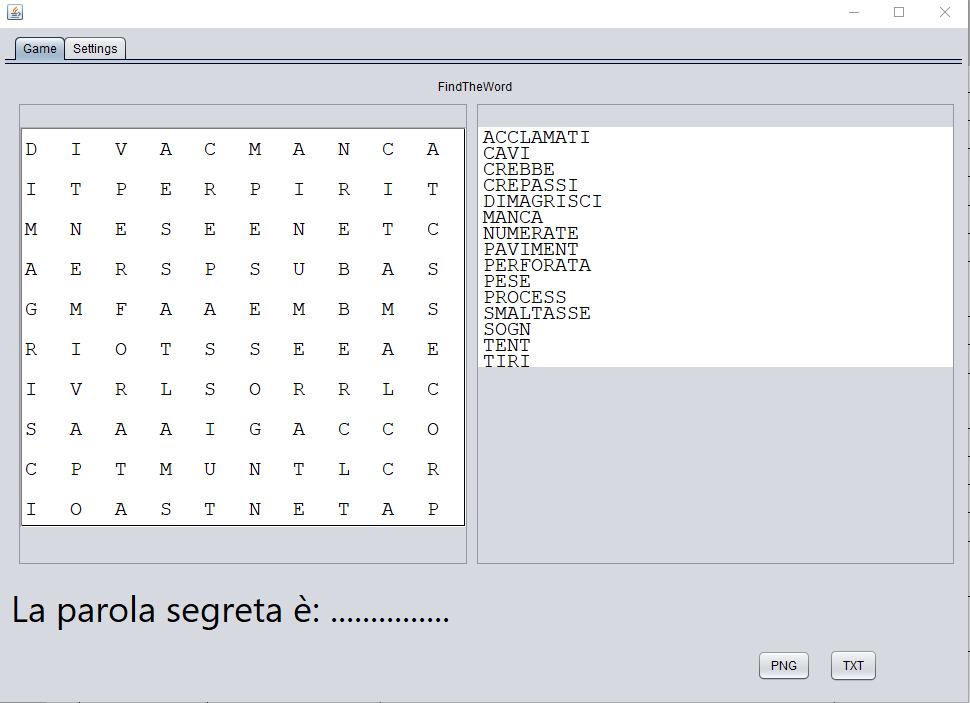


Figura 36 - TC-05

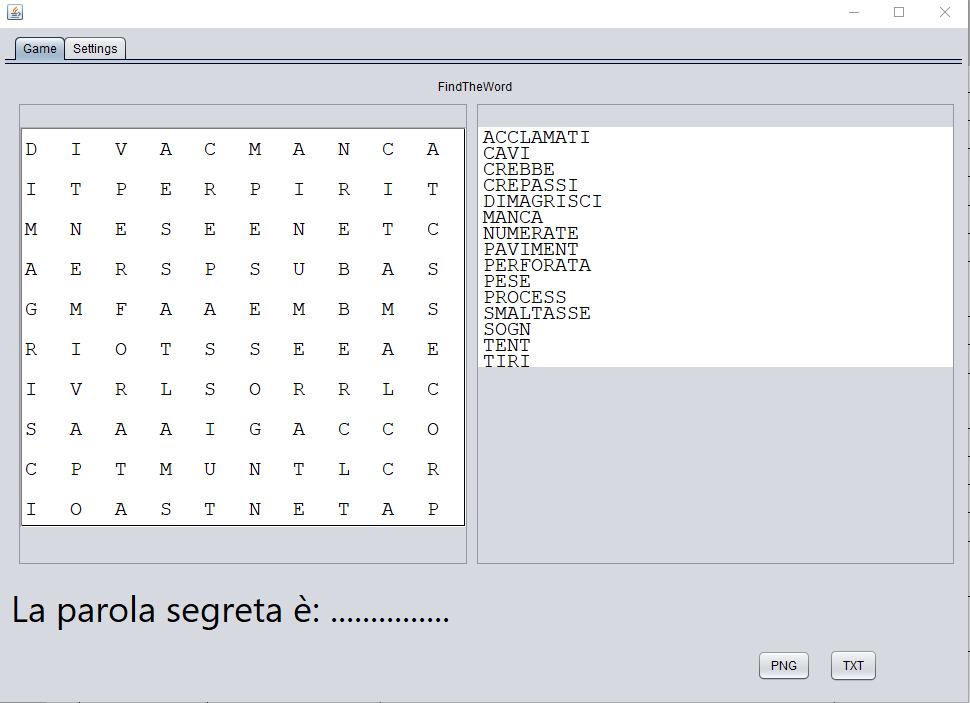


Figura 37 - TC-06



Figura 38 - TC-07 - File Output



Figura 39 - Output Immagine



Figura 40 - Output TXT File

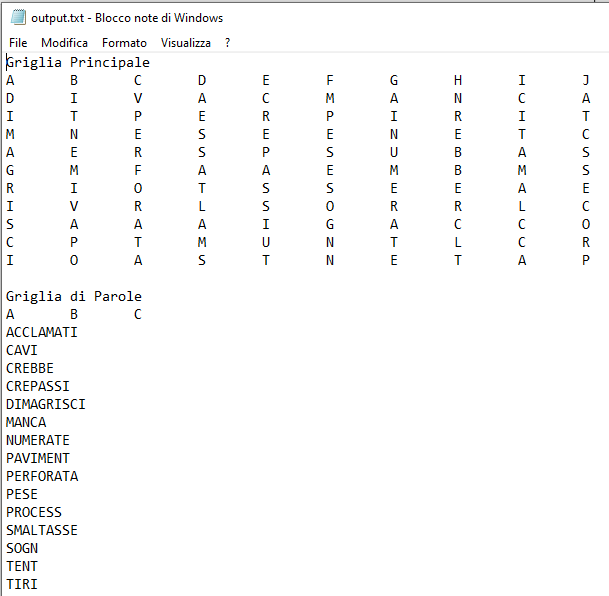


Figura 41 - Output TXT

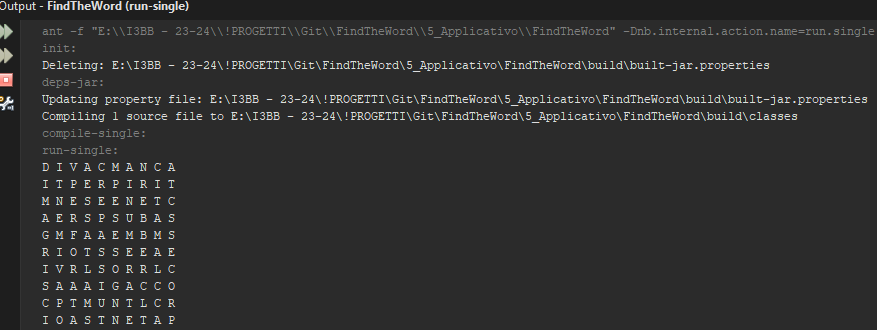


Figura 42 - TC-09

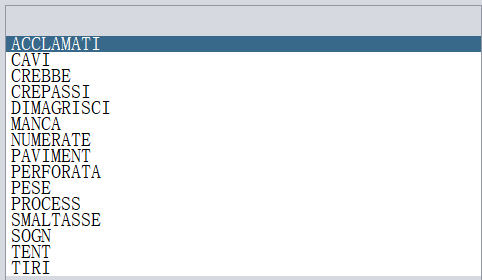


Figura 43 - TC-14

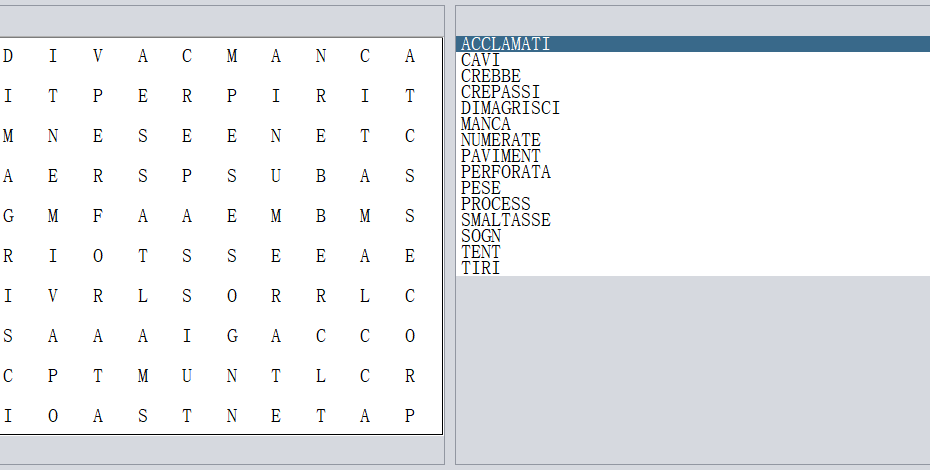


Figura 44 - TC-15,16,17,18

## Mancanze/limitazioni conosciute

**Mancanza di UnitTest:**

Non sono stati scritti UnitTest per il progetto.

Ciò è dovuto alla mancanza di conoscenza e di esperienza nell’implementazione di UnitTest. Tuttavia, si riconosce l’importanza degli UnitTest e si cercherà di implementarli in futuro.

**Mancanza di un dizionario personalizzato:**

Non è stato implementato un dizionario personalizzato nel progetto. Ciò è dovuto al ritardo nel progetto e una limitazione di tempo. Inoltre è stato progettato male il modo in cui si doveva implementare il dizionario.

**Mancanza di multi colonne:**

Non sono state introdotte delle multi colonne per mancanza di tempo e soprattutto per mancanza di progettazione.

Ho dovuto documentarmi molto per trovare il modo di farlo ma non sono riuscito ad implementarle, tuttavia esiste solo una colonna dove ci sono le parole.

# Consuntivo

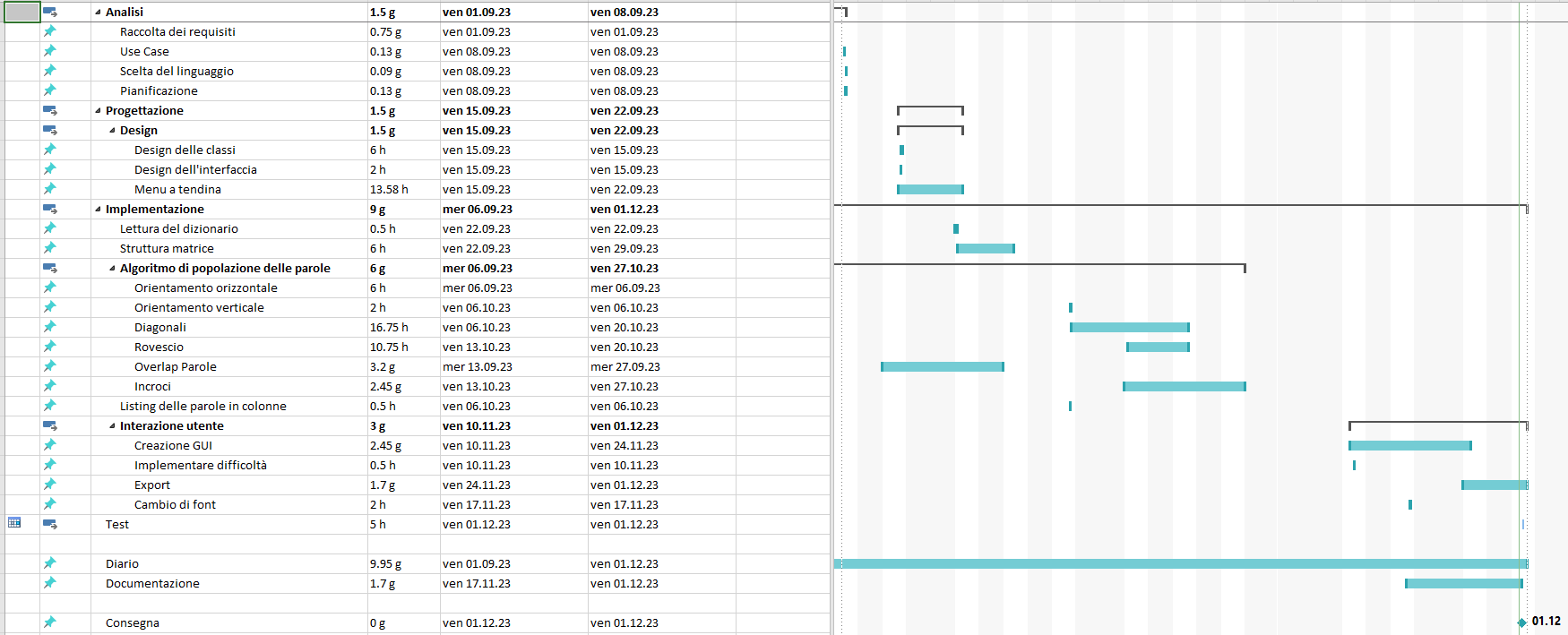


Figura 45 - Gannt Consuntivo

# Conclusioni

Visto che non sono riuscito a trovare online progetti simili, questa applicazione potrebbe attrarre molte persone grazie a un'interfaccia molto semplice e al fatto che non esiste una cosa del genere online.

In conclusione mi sento molto soddisfatto per quello che ho fatto e sono sicuro che potrei migliorarlo nel futuro, perciò questo progetto è riuscito ma con alcuni intoppi e alcuni requisiti funzionali non implementati.

## Sviluppi futuri

* Aggiungere la scelta della dimensione della griglia
* Aggiungere il listing di parole su più colonne
* Aggiungere un dizionario personalizzato
* Migliorare l’estetica
* Ottimizzare il codice

## Considerazioni personali

In conclusione posso dire che questo progetto mi ha insegnato il modo di approcciarmi ad un lavoro di lunga durata e ho imparato le varie fasi di progettazione, documentazione, diari, test, ....

Il diario mi ha aiutato molto, specialmente per tenere traccia di ciò che avevo già fatto e di cosa dovevo fare la settimana successiva.

Il progetto mi ha aiutato a consolidare e apprendere nuove competenze in Java (ad esempio l’uso di interfacce grafiche) e soprattutto mi ha aiutato a capire a meglio come funzionano varie cose in java, come ad esempio la scoperta dei “Collections” che mi ha aiutato a randomizzare l’arrayList delle parole.

Sono complessivamente soddisfatto del risultato finale, poiché sono riuscito a sviluppare la maggior parte dei requisiti che mi ero prefissato inizialmente.

# Glossario

**Classe:** una classe è un modello che descrive la struttura e il comportamento di un oggetto. È definita tramite variabili di istanza, metodi e costruttori.

**Oggetto**: un oggetto è un'istanza di una classe. È una rappresentazione di specifici dati e comportamenti definiti dalla classe.

**Metodi**: i metodi sono delle funzioni definite all'interno di una classe. Possono essere utilizzati per eseguire operazioni specifiche sugli oggetti della classe.

**Variabile**: una variabile è un'area di memoria in cui possono essere memorizzati dati. Possono essere utilizzate per archiviare e manipolare informazioni all'interno di un programma.

**Costruttore**: un costruttore è un metodo speciale che viene chiamato quando un oggetto viene creato da una classe. Viene utilizzato per inizializzare gli attributi dell'oggetto.

**Ereditarietà**: l'ereditarietà è un meccanismo mediante il quale una classe può ereditare attributi e metodi da un'altra classe. Permette di definire una gerarchia di classi e di creare nuove classi basate su quelle esistenti.

**Override**: l'override è un meccanismo mediante il quale una classe figlia può sovrascrivere un metodo ereditato dalla classe genitore, fornendo una propria implementazione.

**Array**: un array è una struttura dati che permette di memorizzare più valori dello stesso tipo. È possibile accedere agli elementi dell'array utilizzando un indice numerico.

**Cicli**: i cicli permettono di eseguire una serie di istruzioni ripetutamente fino a quando non viene soddisfatta una condizione. I cicli più comuni sono il ciclo for e il ciclo while.

**Condizioni**: le condizioni sono espressioni che restituiscono un valore booleano (vero o falso). Vengono utilizzate per prendere decisioni all'interno di un programma.

**Eccezioni**: un'eccezione è un'evenienza anomala che si verifica durante l'esecuzione di un programma. Viene utilizzata per gestire errori o situazioni impreviste.

**Java**: In informatica Java è un linguaggio di programmazione ad alto livello, orientato agli oggetti e a tipizzazione statica, che si appoggia sull'omonima piattaforma software di esecuzione, specificamente progettato per essere il più possibile indipendente dalla piattaforma hardware di esecuzione.

**GUI:** L'interfaccia grafica, nota anche come GUI in informatica è un tipo di interfaccia utente che consente l'interazione uomo-macchina in modo visuale utilizzando rappresentazioni grafiche

# Bibliografia

## Sitografia

[**https://stackoverflow.com/**](https://stackoverflow.com/)**,** StackOverflow

<https://www.w3schools.com/>, W3Schools

<https://www.javatpoint.com/>, JavaPoint

<https://www.geeksforgeeks.org/>, GeeksForGeeks

<https://www.freecodecamp.org/>, FreeCodeCamp

<https://howtodoinjava.com/>, HowToDoInJava

# Indice delle figure

[Figura 1 - Use Case 7](#_Toc152332335)

[Figura 2 - Gannt Preventivo 8](#_Toc152332336)

[Figura 3 - Organizzazione dei dati 10](#_Toc152332337)

[Figura 4 - Design delle interfacce 11](#_Toc152332338)

[Figura 5 - Design delle interfacce - finale 12](#_Toc152332339)

[Figura 6 - Interfccia all'avvio 13](#_Toc152332340)

[Figura 7 - Interfaccia generata 13](#_Toc152332341)

[Figura 8 - Interfaccia settings 14](#_Toc152332342)

[Figura 9 - Design procedurale 15](#_Toc152332343)

[Figura 10 - Design algoritmo 16](#_Toc152332344)

[Figura 11 - Classe Consts 17](#_Toc152332345)

[Figura 12 - Metodo CheckOrientation 18](#_Toc152332346)

[Figura 13 - Metodo isInBounds 19](#_Toc152332347)

[Figura 14 - isValidOverlapping 20](#_Toc152332348)

[Figura 15 - Metodo insertWord 21](#_Toc152332349)

[Figura 16 - Metodo setWord 22](#_Toc152332350)

[Figura 17 - Metodo setWordsWhileYouCan 22](#_Toc152332351)

[Figura 18 - Classe GridModel 23](#_Toc152332352)

[Figura 19 - Classe WordsTableModel 24](#_Toc152332353)

[Figura 20 - Interfaccia MainGame 25](#_Toc152332354)

[Figura 21 - Interfaccia Settings 26](#_Toc152332355)

[Figura 22 - Interfaccia Game Avviata 26](#_Toc152332356)

[Figura 23 - Evento Bottone PNG 27](#_Toc152332357)

[Figura 24 - Metodo captureComponent 27](#_Toc152332358)

[Figura 25 - Evento Selezione Font 27](#_Toc152332359)

[Figura 26 - Metodo applyFontToJTable 28](#_Toc152332360)

[Figura 27 - Evento TXTActionPerformed 28](#_Toc152332361)

[Figura 28 - Metodo exportToTxt 29](#_Toc152332362)

[Figura 29 - Metodo writeTableToTxt 29](#_Toc152332363)

[Figura 30 - Evento Bottone Genera 30](#_Toc152332364)

[Figura 31 - Gannt Consuntivo 41](#_Toc152332365)