WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA
z siedzibą w Rzeszowie

KOLEGIUM INFORMATYKI STOSOWANEJ

Kierunek: INFORMATYKA

Specjalność: Programowanie

Maksym Diakiv
Nr albumu studenta w68138

Gra Sudoku

Promotor: dr. Marek Giebułtowski

PROJEKT

Rzeszów 2024

Spis treści

1 Wprowadzenie **3**
1.1 Cel projektu 3

2 Wymagania Funkcjonalne **4**
2.1 Główne funkcje 4
2.1.1 Generacja gry 4
2.1.2 Zapis stanu gry 4

3 Opis Klas i Metod **5**
3.1 SudokuBuildLogic 5
3.2 GameGenerator 5
3.3 GameLogic 5
3.4 SudokuSolver 6
3.5 SudokuUtilities 6
3.6 LocalStorageImpl 6
3.7 SudokuGame 6
3.8 UserInterfaceImpl 7

4 Działanie Projektu **8**
4.1 Uruchomienie 8
4.2 Koniec gry 8

Literatura **10**

Rozdział 1

Wprowadzenie

1.1 Cel projektu

Celem projektu jest tworzenie aplikacji, która zezwoli grać w Sudoku. Jesto to gra, w której musimy sprawdzać czy liczba znajduje się w liniach poziomych i pionowych i w kwadracie w którym znajduje się liczba. Celem gry jest uzupełnienie wszystkich pól liczbami od 1 do 9.

Rozdział 2

Wymagania Funkcjonalne

2.1 Główne funkcje

2.1.1 Generacja gry

a) Generacja pól.

b) Sprawdzanie czy wszystko zostało zgenerowane poprawnie.

c) W przypdaku poprawnego uzupełnienia pól, wyświetlanie wiadomosci i generacja nowej gry.

2.1.2 Zapis stanu gry

a) W przypadku zamknięcia gry do uzupełnienia wszystkich pól, wszystkie wartości zostaną zapisane.

b) Przy uruchomienu projektu odbędzie się odczyt ostatniej gry..

Rozdział 3

Opis Klas i Metod

3.1 SudokuBuildLogic

- build - Próbuje załadować istniejące dane gry. Jeśli to się nie powiedzie, tworzy nową grę i zapisuje ją w lokalnym magazynie. Ustawia kontroler dla interfejsu użytkownika. Aktualizuje interfejs użytkownika, aby wyświetlić aktualny stan gry.

3.2 GameGenerator

Klasa GameGenerator jest odpowiedzialna za generowanie plansz Sudoku, w tym zarówno kompletnych (rozwiązanych), jak i niekompletnych (z pustymi polami).

- getNewGameGrid() - generuje nową planszę Sudoku.
- getSolvedGame() - tworzy pełną planszę, przestrzegając zasad Sudoku.
- unsolveGame() - suwa losowo 40 wartości z pełnej planszy, tworząc niekompletną planszę do rozwiązania.
- clearArray() - resetuje planszę, ustawiając wszystkie jej wartości na 0. Używana jest do resetowania planszy Sudoku w przypadku, gdy proces generowania rozwiązanego układu nie przebiega pomyślnie po wielu próbach.

3.3 GameLogic

Klasa GameLogic jest odpowiedzialna za logikę gry Sudoku, w tym generowanie nowej gry oraz sprawdzanie poprawności i stanu planszy.

- getNewGame() - tworzy nową grę Sudoku.
- checkForCompletion() - sprawdza, czy plansza Sudoku jest rozwiązana, czy nadal jest aktywna (niekompletna).
- tilesAreNotFilled() - sprawdza, czy na planszy Sudoku znajdują się puste pola (wartość 0).
- sudokuIsInvalid() - sprawdza, czy plansza Sudoku jest nieprawidłowa.
- squaresAreInvalid() - sprawdza, czy którykolwiek z kwadratów 3x3 na planszy jest nieprawidłowy. Dzieli planszę na trzy rzędy kwadratów: górny, środkowy i dolny, i sprawdza każdy z nich za pomocą rowOfSquaresIsInvalid.

- rowOfSquaresIsInvalid() - sprawdza, czy którykolwiek z kwadratów w danym rzędzie kwadratów (górny, środkowy, dolny) jest nieprawidłowy. Wywołuje funkcję squareIsInvalid dla każdego kwadratu w rzędzie.

- squareIsInvalid() - sprawdza, czy dany kwadrat 3x3 na planszy Sudoku jest nieprawidłowy

- columnsAreInvalid() - sprawdza, czy którakolwiek kolumna na planszy Sudoku jest nieprawidłowa.

- rowsAreInvalid() - sprawdza, czy którykolwiek wiersz na planszy Sudoku jest nieprawidłowy.

- collectionHasRepeats() - sprawdza, czy dana kolekcja zawiera powtórzenia.

3.4 SudokuSolver

Klasa SudokuSolver implementuje algorytm rozwiązywania Sudoku, znany jako metoda z powrotem.

- puzzleIsSolvable() - sprawdza, czy dana łamigłówka Sudoku jest rozwiązywalna.

- typeWriterEnumerate() - enumeruje wszystkie puste komórki na planszy Sudoku.

3.5 SudokuUtilities

Klasa SudokuUtilities zapewnia pomocnicze metody do manipulacji tablicami używanymi w grze Sudoku.

- copySudokuArrayValues - kopiuje wartości z jednej dwuwymiarowej tablicy Sudoku do innej.

- copyToNewArray - tworzy i zwraca nową dwuwymiarową tablicę z wartościami skopiowanymi z podanej tablicy.

3.6 LocalStorageImpl

Klasa LocalStorageImpl implementuje interfejs IStorage, który zawiera metody do zapisywania i odczytywania danych gry Sudoku z lokalnego systemu plików.

- updateGameData - zapisuje aktualny stan gry Sudoku do pliku.

- getGameData() - Ta metoda odczytuje stan gry Sudoku z pliku.

3.7 SudokuGame

Klasa SudokuGame reprezentuje stan gry Sudoku i jest zaprojektowana jako klasa niezmienna.

- getGameState() - zwraca aktualny stan gry.

- getCopyOfGridState() - zwraca głęboką kopię dwuwymiarowej tablicy gridState.

3.8 UserInterfaceImpl

Klasa UserInterfaceImpl jest implementacją interfejsu IUserInterfaceContract.View oraz rozszerza klasę EventHandler<KeyEvent>, co umożliwia obsługę zdarzeń klawiatury w interfejsie użytkownika gry Sudoku.

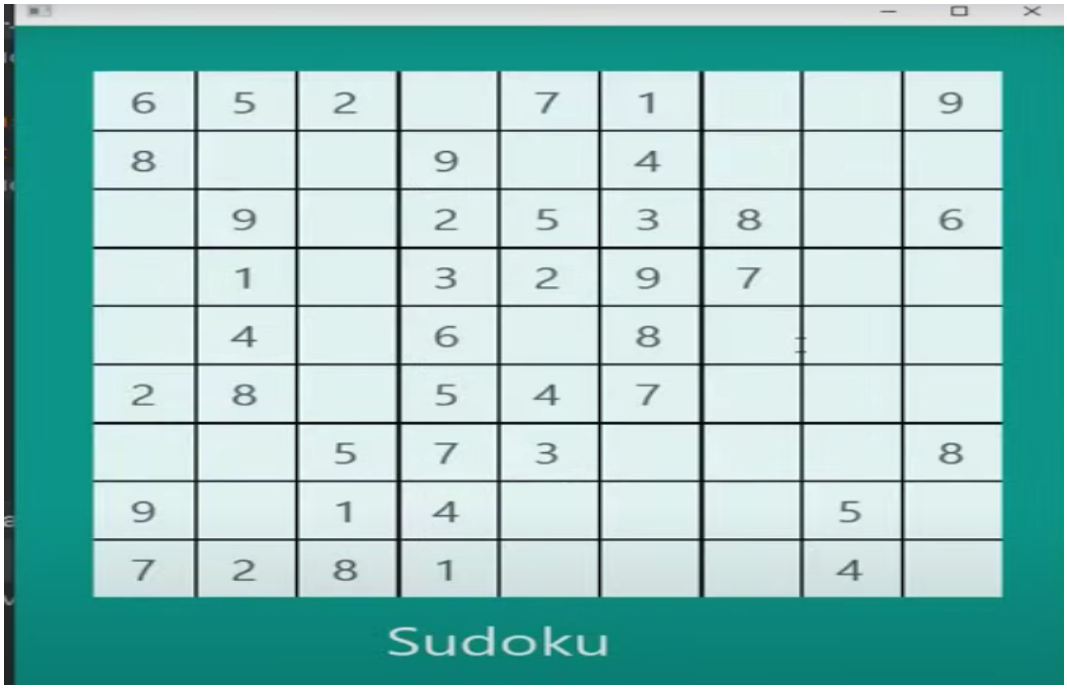
- initializeUserInterface() - inicjalizuje interfejs użytkownika przez rysowanie tła, tytułu, planszy Sudoku, pól tekstowych oraz linii siatki.
- drawTextFields - rysuje pola tekstowe.
- drawGridLines - rysuje linie siatki Sudoku.
- drawBackground - rysuje tło głównego okna aplikacji.
- drawSudokuBoard - rysuje tło planszy Sudoku.
- drawTitle - rysuje tytuł gry Sudoku na interfejsie użytkownika.
- styleSudokuTile - Metoda pomocnicza do stylowania pojedynczego pola Sudoku w interfejsie użytkownika, ustawiająca font, rozmiar, kolor tła i pozycję.

Rozdział 4

Działanie Projektu

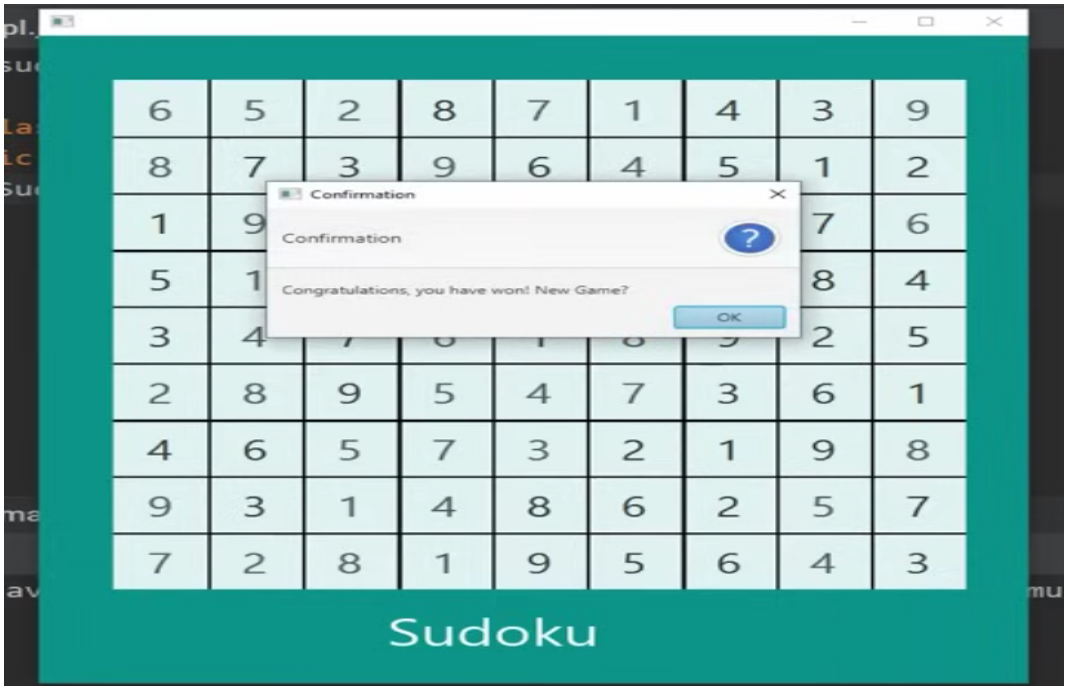
4.1 Uruchomienie

Po uruchomieniu projektu dostajemy płanszę Sudoku, którą musimy uzupełnić.



4.2 Koniec gry

Po poprawnym uzupełnieniu wszystkich pól dostajemy wiadomość o końcu i możemy zgenerować mową grę


6 5 2 8 7 1 4 3 9
8 7 3 9 6 4 5 1 2
1 9 7 6
5 1 8 4
3 4 2 5
2 8 9 5 4 7 3 6 1
4 6 5 7 3 2 1 9 8
9 3 1 4 8 6 2 5 7
7 2 8 1 9 5 6 4 3
Confirmation
Confirmation
Congratulations, you have won! New Game?
OK
Sudoku

.

Bibliografia

[1] https://openjfx.io/

[2] https://www.freecodecamp.org/

[3] https://docs.oracle.com/en/java/