Escape The Maze

Specyfikacja Implementacji

Maksym Andrushchenko

14 kwietnia 2025

Spis treści

1 Wstęp

1.1 Cel dokumentu

Dokument opisuje implementację gry Labirynt, w tym jej strukturę kodu, algorytmy i architekturę systemu.

1.2 Zakres implementacji

Implementacja obejmuje generowanie labiryntu, mechanikę gry, interakcję użytkownika oraz zapis wyników.

1.3 Technologie i narzędzia

Gra będzie napisana w języku Java, a do GUI wykorzystana zostanie biblioteka JavaFX.

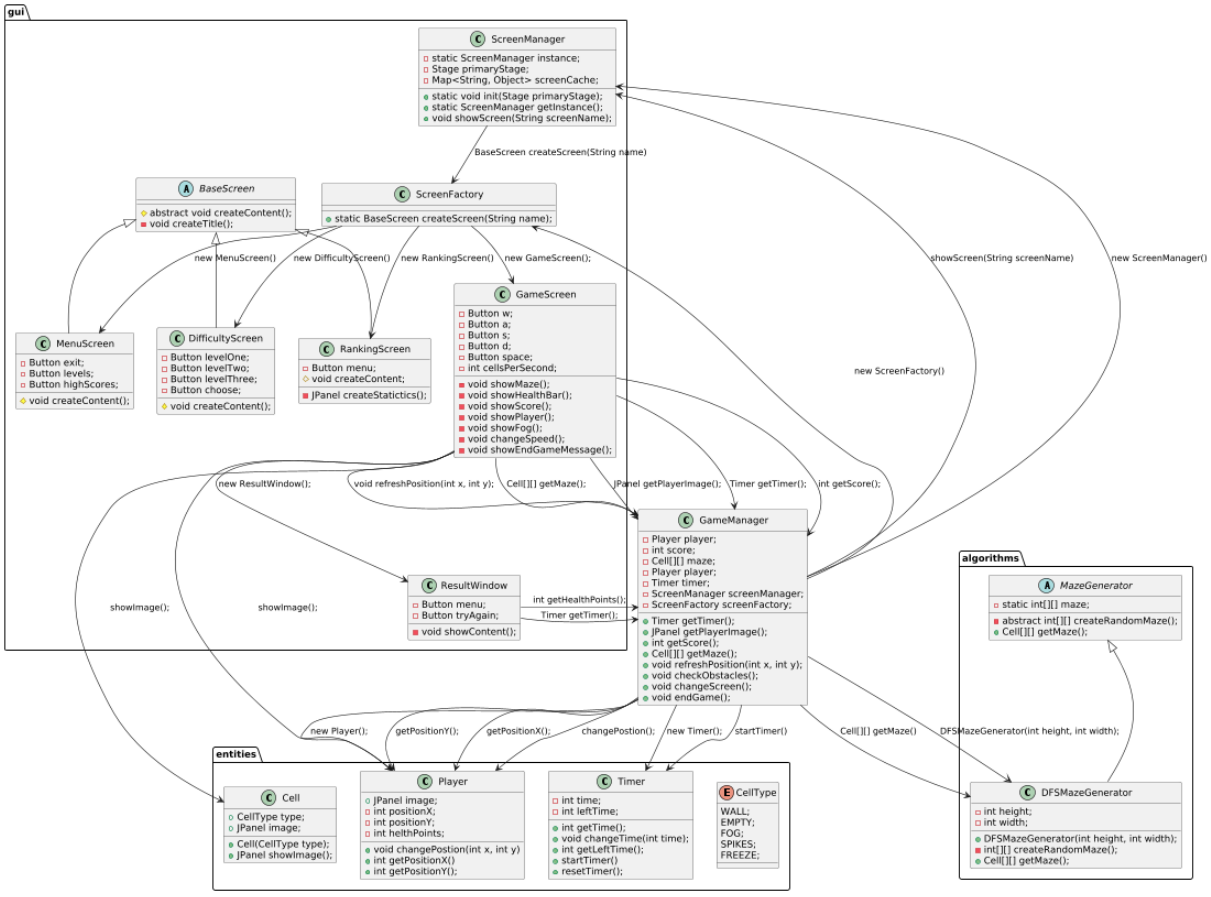
2 Planowanie gry

2.1 Podział na moduły

Gra składa się z takich modułów:

- algorytmów;
- gui (interfejs użytkownika);
- encji;

2.2 Diagram klas



*2*2

Rysunek 1: Diagram UML wygenerowany z PlantText

3 Implementacja funkcjonalności

3.1 System gry i interfejs użytkownika

3.1.1 BaseScreen.java

Opis: Abstrakcyjna klasa bazowa dla ekranów menu (MenuScreen, DifficultyScreen, RankingScreen). Odpowiada za wyświetlanie nagłówka z nazwą gry "Escape the Maze!". Zawiera metodę abstrakcyjną `createContent()`, która tworzy zawartość dolnej części ekranu (przyciski, opisy).

3.1.2 MenuScreen.java

Opis: Główne menu gry, pozwalające rozpocząć grę, przejść do rankingu lub zakończyć aplikację. Dziedziczy po BaseScreen.java.
Ma przyciski:

- "START": zmiana widoku na DifficultyScreen.java;
- "RANKING": zmiana widoku na RankingScreen.java;
- "EXIT": zakończenie gry.

3.1.3 DifficultyScreen.java

Opis: Ekran wyboru poziomu trudności (easy, normal, hard, impossible). Dziedziczy po BaseScreen.java.
Ma przyciski:

- "Level x": wybiera jeden poziom trudności, każdy przycisk ma krótki opis poziomu;
- "CHOOSE": zaczyna grę z odpowiednie wybranym poziomem trudności i zmienia widok na GameScreen.java.

3.1.4 RankingScreen.java

Opis: Wyświetla najlepsze wyniki graczy i pozwala wrócić do menu głównego. Dziedziczy po BaseScreen.java. Odczytuje wyniki z pliku score.txt.
Ma przycisk:

- "MENU": powrót do widoku z menu (MenuScreen.java).

3.1.5 GameScreen.java

Opis: Główny ekran gry. Wyświetla labirynt, gracza, pasek życia, punkty i timer. Obsługuje sterowanie i aktualizuje widok. Każdy nazwany element jest pobierany od GameMenager.java. Obsługa sterowania odbywa się po wciśnięciu odpowiedniego przycisku. Po zmianie pozyscji wywołuje metodę refreshPosition() w klasie GameManager i podalsza obsługa jest sterowana GameManagerem. Po zakończeniu gry (brak życia, koniec czasu, wygrana) uruchamiany ResultWindow.

3.1.6 ResultWindow.java

Opis: Ekran końcowy. Informuje gracza o zakończeniu gry: wygrana, przegrana lub brak czasu. Wyświetla punkty i informację o nowym rekordzie. Odczytuje zdrowie i czase z `GameManager` – wybiera odpowiedni komunikat. Wyświtla zdobyte punkty. Sprawdza rekord w pliku score.txt, aktualizacja przy nowym wyniku.
Ma przyciski:

- "MENU": powrót do widoku z menu (MenuScreen.java).
- "TRY AGAIN": zaczyna grę z odpowiednie wybranym poziomem trudności ponownie.

3.1.7 ScreenManager i ScreenFactory

Opis: Odpowiadają za zarządzanie ekranami gry. Zastosowano wzorce projektowe: Singleton i Factory Method.
ScreenManager przechowuje instancję Stage oraz mapę ekranów. Dla ekranów typu "GAME"tworzy GameScreen niezależnie (nie dziedziczy po BaseScreen). Dla pozostałych ekranów wykorzystuje ScreenFactory, która tworzy odpowiedni BaseScreen. Po utworzeniu ekran jest umieszczany w nowej scenie i wyświetlany.

3.2 GameManager

Opis: Klasa centralna zarządzająca logiką gry. Obsługuje inicjalizację gry, gracza, interakcje, punkty i stan gry. Przechowuje tablicę Cell[][] reprezentującą labirynt. Zawiera instancję klasy Player, Timer i punkty. Obsługuje logikę ruchu gracza, kolizji, zbierania monet, efektów przeszkód. Odpowiada za zakończenie gry.

3.3 Entities (encje)

3.3.1 Player.java

Opis: Reprezentuje gracza w grze. Przechowuje informacje o stanie zdrowia, pozycji i wyglądzie zależnie od efektu.

3.3.2 Cell.java

Opis: Klasa reprezentująca jedną komórkę labiryntu. Zawiera informację o tym jaką komórką jest (zmienna typu CellType), metodę zwracającą ten typ i metodę która zwraca panel z odpowiednim zdjęciem.

3.3.3 Enum CellType

Opis: Ma wartości dla każdego rodzaju komórki.

3.3.4 Timer.java

Opis: Klasa licznik czasu. Liczy czas gry w osobnym wątku, może być zresetowany i odczytywany przez GameManager. Można zmieniać czas do którego liczy.

3.4 Algorithms (algorytmy)

3.4.1 MazeGenerator

Opis: Klasa absrtakcyjna która odpowiada za szkielet klasy generującej labirynt. Można po niej dziedziczyć i tworzyć losowy labirynt. Przechowuje labirynt w typie int[][] ale zwraca publicznie w typie Cell[][].