Specyfikacja implementacyjna

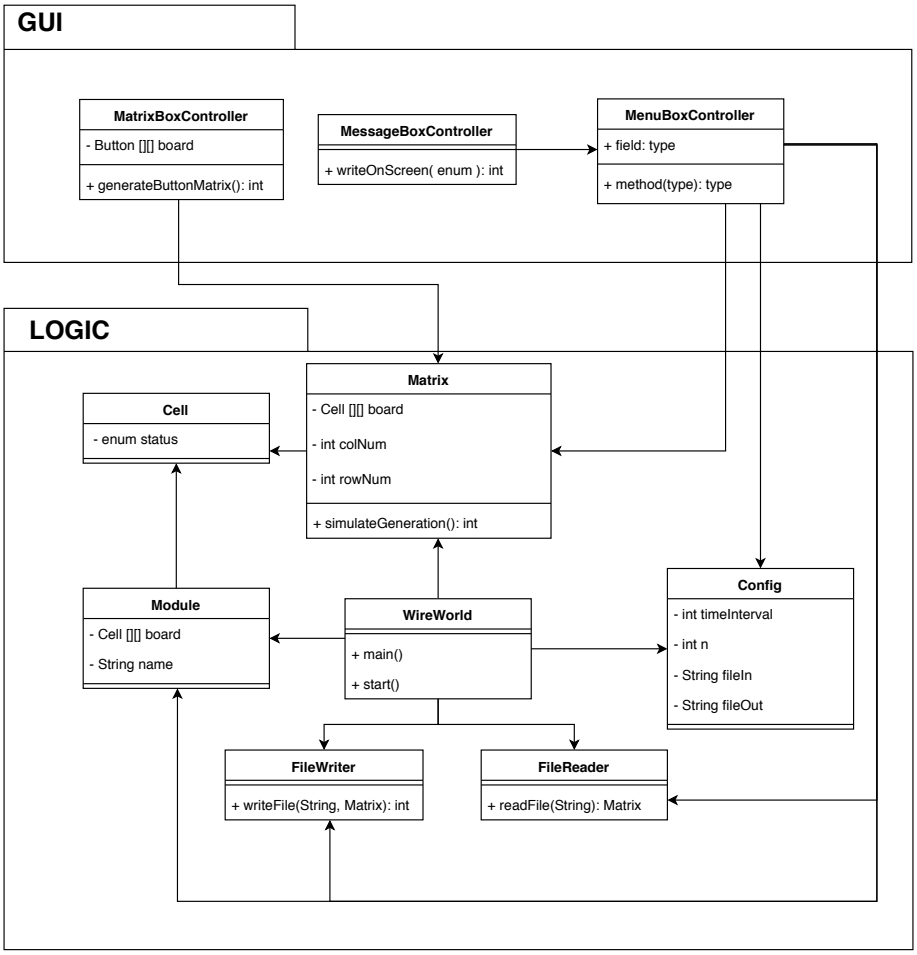
Karolina Czachorska, Piotr Ferdynus

21.05.2019

Spis treści

1 Diagram klas

Diagram modułów programu *WireWorldSimulator2000* przedstawia się następująco:



2 Opis poszczególnych klas

2.1 MatrixBoxController

Klasa ta należy do pakietu GUI, który odpowiada za graficzny interfejs użytkonika. Zajmuje się ona obsługą planszy wyświetlanej na ekranie. Jej metoda generateButtonMartix tworzy panszę składającą się z przycisków, które umożliwią użytkownikowi wybór rodzaju komórki poprzez kliknięcie odpowiednią liczbę razy.

2.2 MessageBoxController

Klasa odpowiada za obsługę graficznego okna komunikatów, wyświetlającego status czynności programu, ewentualne błędy, ostrzeżenia i sugestie.

2.3 MenuBoxController

Klasa ta służy do interakcji użytkownika z programem poprzez wybór odpowiednich opcji takich jak: wczytanie z pilku, zapis do pliku, start i stop programu.

2.4 Cell

Klasa przechowuje stan komórki za pomocą typu *enum*. Pozwala na ustalenie oraz odczytanie jej stanu.

2.5 Matrix

Klasa reprezentująca planszę, która składa się z tablicy obiektów typu Cell, określonej liczby wierszy i kolumn. Posiada ona metodę simulateGeneration która tworzy jedną nową generację planszy w oparciu o poprzednią wykorzystując zasady Wireworld.

2.6 Config

Klasa przechowuje zmienne odpowiedzialne za konfigurację pracy programu.

- timeInterval – minimalny interwał czasowy między pokoleniami
- n – liczba generacji do symulowania (przy skończonym trybie symulacji)
- fileIn – nazwa pliku wejściowego przy wczytywaniu danych z pliku
- fileOut – nazwa pliku wyjściowego przy zapisie danych do pliku

2.7 WireWorld

Klasa odpowiedzialna za uruchomienie programu wraz z środowiskiem graficznym. Obsługuje symuluację, steruje jej przebiegiem, zarządza ewentualnymi błędami.

2.8 FileWriter

Klasa odpowiedzialna za zapis do pliku. Jej metoda writeFile zapisuje Matix do pliku o podanej nazwie. Zwrócona wartość int informuje o tym czy zapis przebiegł poprawnie.

2.9 FileReader

Klasa odpowiedzialna za odczyt z pliku. Jej metoda readFile czyta plik i tworzy Matix lub Module z pliku o podanej nazwie. Zwrócona wartość int informuje o tym czy odczyt przebiegł poprawnie.

2.10 Module

Klasa odpowiedzialna za przechowanie gotowych modułów w programie. Przechowywane są one jako tablica obiektów Cell.

3 Opis głównych algorytmów

Za pomocą przycisków klasy *MatrixBoxController* lub po wczytaniu z pliku użytkownik ustala stan komórek siatki. Po wybraniu trybu symulacji, interwałów czasowych oraz rozpoczęciu działania przyciskiem "START" program rozpoczyna symulację pokoleń. Macierz przechowywanajest w klasie *Matrix*, za symulację odpowiada metoda *simulateGeneration*, która zwraca wartość całkowitą, 0 w przypadku powowdzenia i różną od zera w przypadku wystąpienia błędu. Następnie jest ona wyświetlana na ekranie za pomocą klasy *MatrixBoxController*. Klasa *WireWorld* kontrolująca przebieg symulacji, powtarza ten cykl aż do spełnienia warunku liczby wygenerowanych planszy lub do naciśnięcia przycisku "STOP".

4 Testy

4.1 Test klasy FileWriter

Dla obiektu Matrix o ustalonym wypełnieniu wartościami:

```
[EMPTY, TAIL, HEAD, COND,
TAIL, EMPTY, TAIL, HEAD,
TAIL, HEAD, COND, EMPTY,
EMPTY, COND, COND, EMPTY]
```

Zapisany plik powinien wyglądać następująco:

```
0 2 3 1
2 0 2 3
2 3 1 0
0 1 1 0
```

4.2 Test klasy FileReader

Dla następującego pliku:

```
0 0 3 1
2 2 1 3
1 2 3 0
0 0 3 2
```

Obiekt Matrix powinien mieć następujące wypełnienie:

```
[EMPTY, EMPTY, HEAD, COND,
TAIL, TAIL, COND, HEAD,
COND, TAIL, HEAD, EMPTY,
EMPTY, EMPTY, HEAD, TAIL]
colNum=4
rowNum=4
```

4.3 Test klasy Matrix

Początkowy obiekt Matrix:

```
EMPTY COND EMPTY
HEAD EMPTY COND
EMPTY COND EMPTY
```

Obiekt Matrix po jednym pokoleniu:

```
EMPTY HEAD EMPTY
TAIL EMPTY COND
EMPTY HEAD EMPTY
```