Politechnika Warszawska
Wydział Elektryczny

Sprawozdanie „WireWorld”

Autorzy:
Grzegorz Kopyt
Daniel Sporysz

3 czerwca 2018

Spis treści

1 Opis działania

Program jest implementacją automatu komórkowego opartego na regułach „gry w życie” Johna Conwaya w wariancie „WireWorld”.

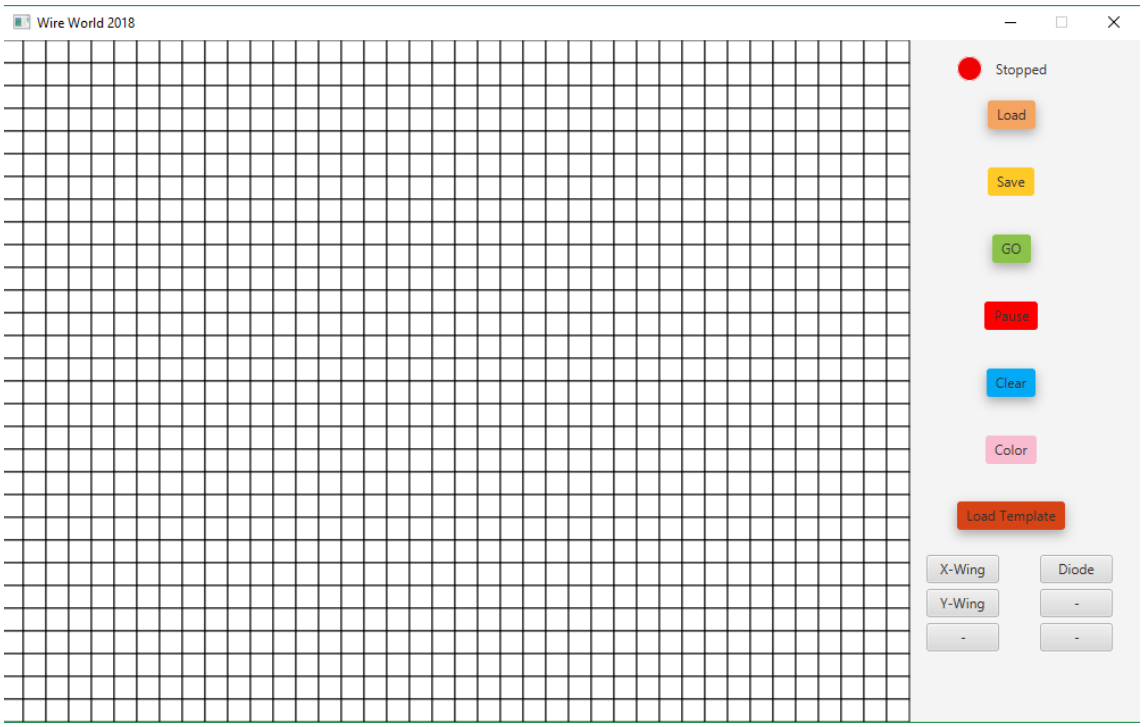
Za pomocą interfejsu graficznego program przedstawia zmiany pól jakie zachodzą na planszy zgodnie z zasadami gry.

Pracę programu można konfigurować na kilka sposobów: wczytać początkową konfigurację planszy z pliku graficznego, stworzyć własną planszę za pomocą narzędzi do edycji pól lub w sposób mieszany, czyli wczytując konfigurację z pliku i dalsza jego edycja za pomocą edytora programu.

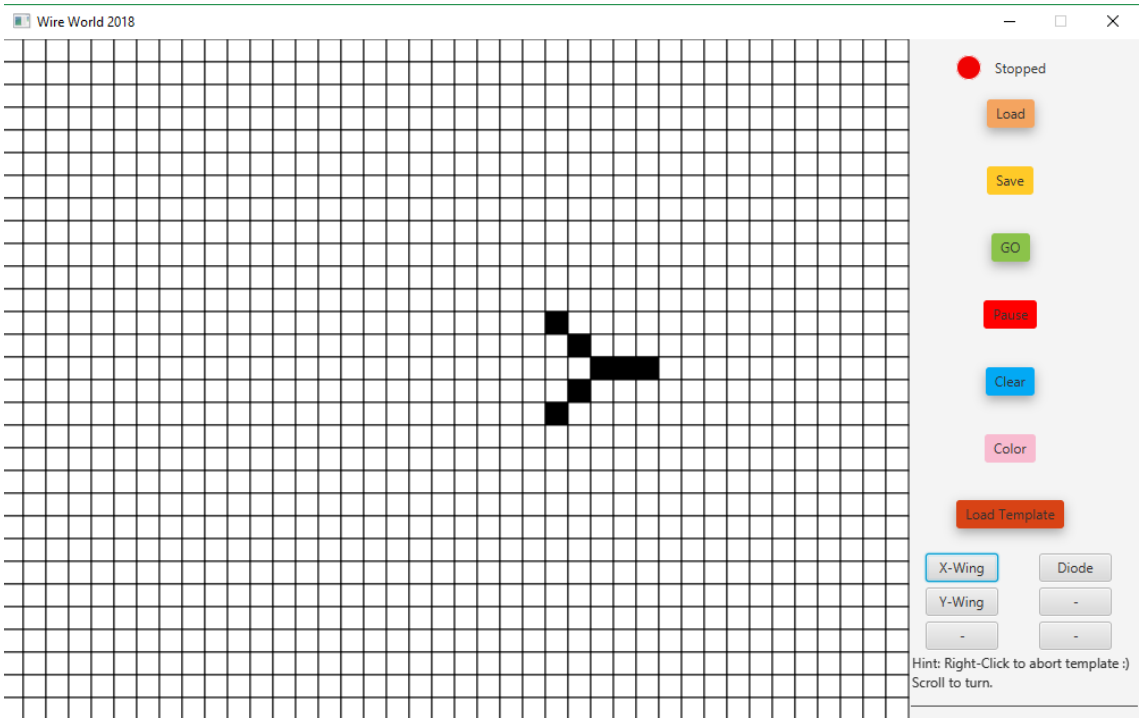
Analizę i modyfikację planszy można przerwać lub wznowić w każdej chwili pracy programu. Gdy generacja jest wstrzymana, użytkownik może ręcznie zmodyfikować planszę lub zapisać jej obecny stan do pliku graficznego.

2 Możliwości programu

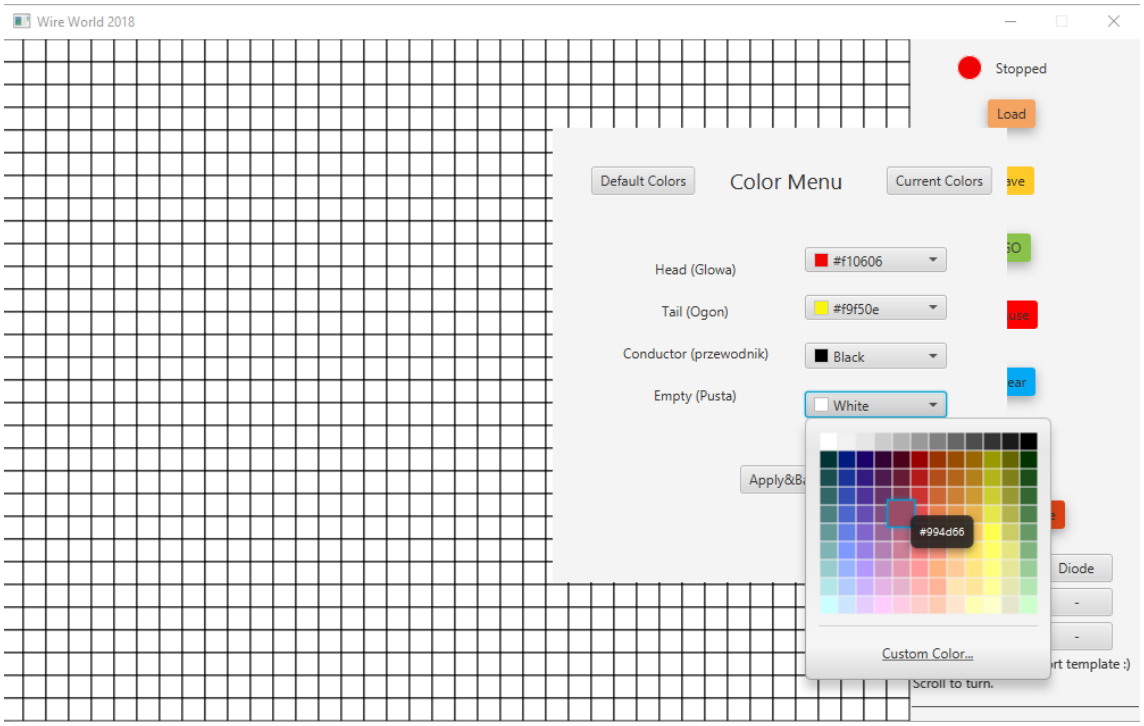
2.1 Edycja planszy



- Zmiana koloru pól poprzez "przeciągnięcie" - zmieniają się w kolejności pusty->przewodnik->ogon->głowa.
- Zmiana koloru pól poprzez "kliknięcie" lewym przyciskiem myszy - zmieniają się w kolejności pusty->przewodnik->ogon->głowa.
- Zmiana koloru pól poprzez "kliknięcie" prawym przyciskiem myszy - nadanie koloru pusty.
- Wstawianie gotowych wzorów na plansze poprzez naciśnięcie jednego z przycisków w prawym dolnym rogu i przesuwając kursorem na planszy umieszczenie go w wybranym miejscu i zatwierdzenie lewym przyciskiem myszy lub porzucenie prawym (możliwość obrotu elementów poprzez "scroll").

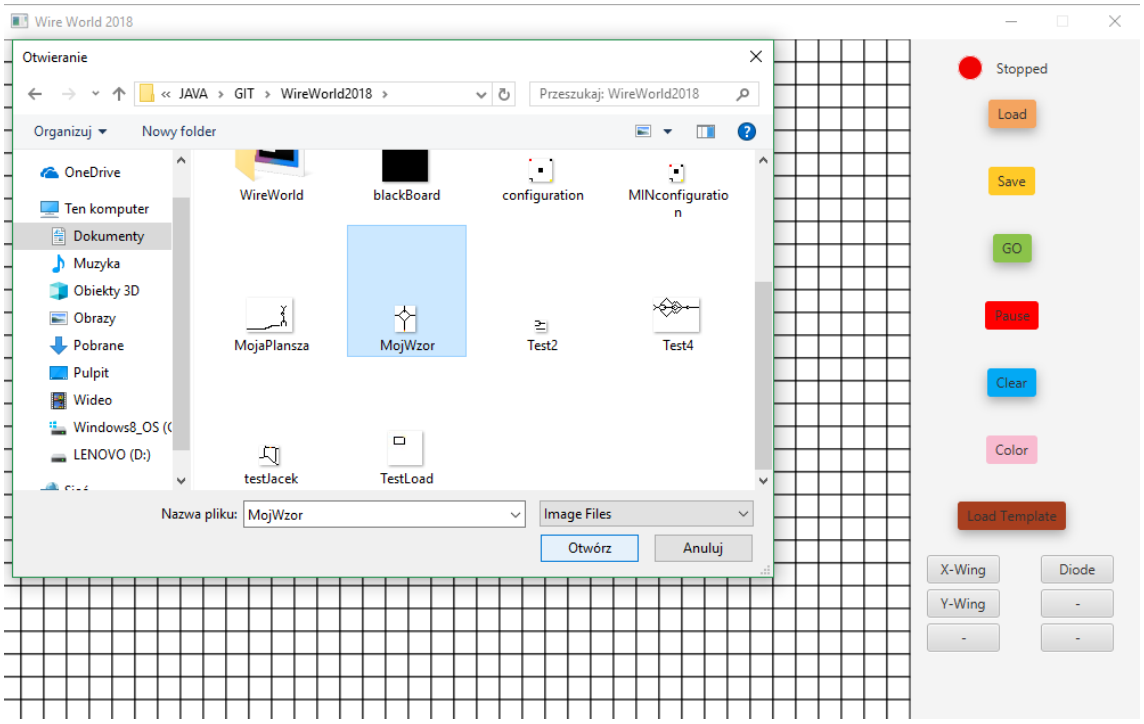


2.2 Zmiana domyślnych kolorów planszy

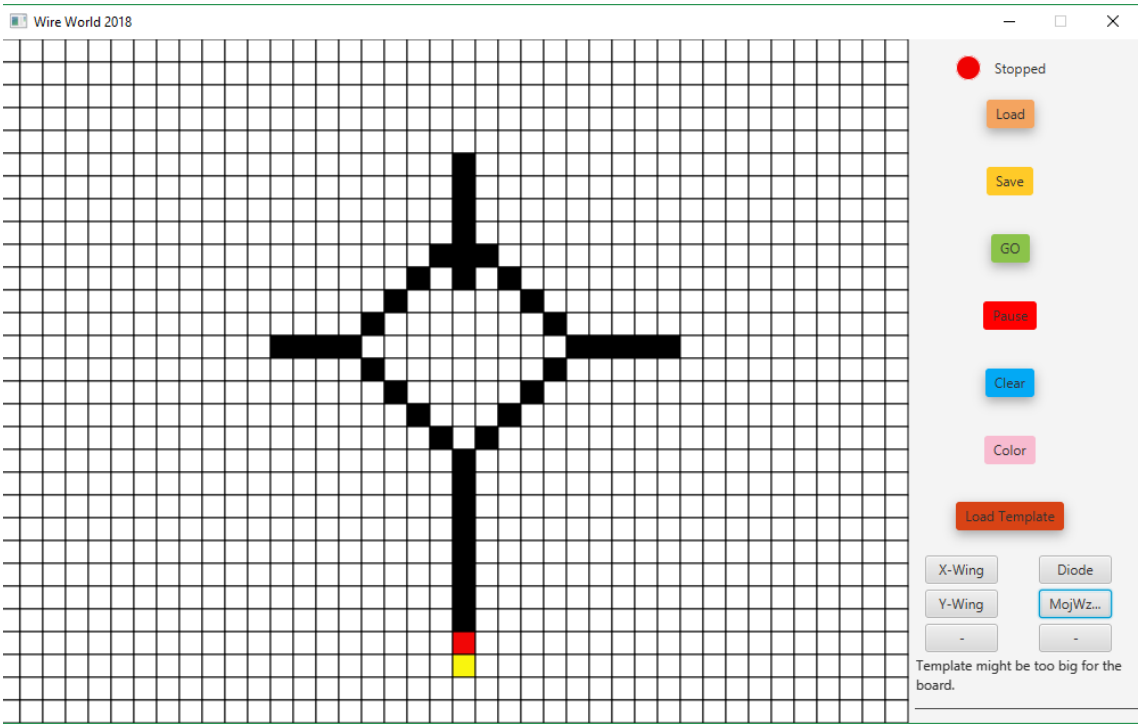


- W wygodnym meny koloru można wybrać własne kompozycje kolorów.

2.3 Dodawanie własnych wzorów



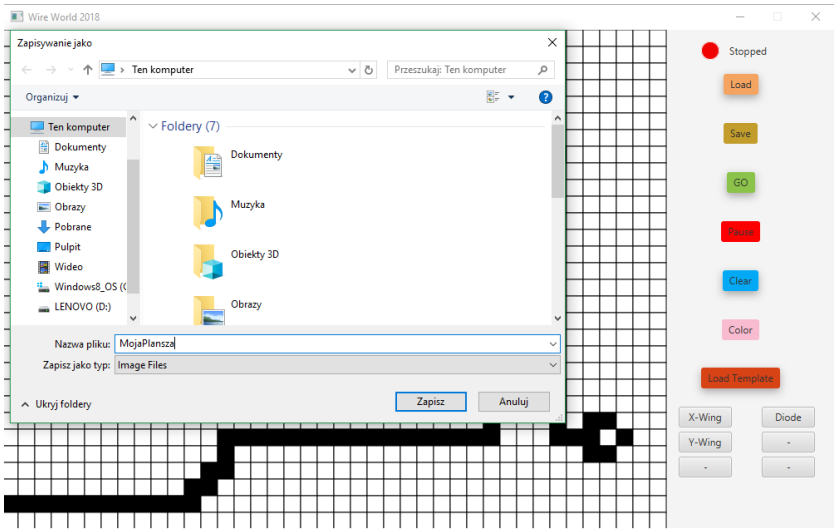
- Możliwość dodania trzech wzorów poprzez przycisk ”LoadTemplate”
- Nazwa pliku ze wzorem pojawi się na jednym z przycisków w prawym dolnym rogu.



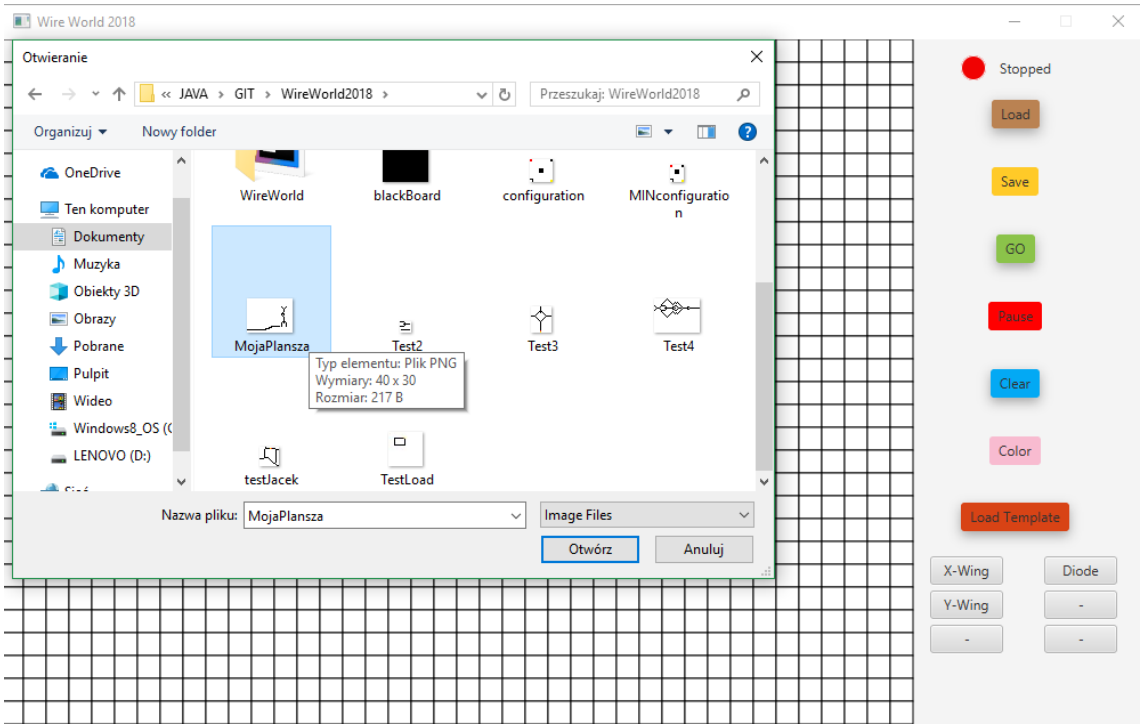
- Plik musi być w formacie graficznym (.png, .jpeg), pokolorowany kolorami biały(pusty), czarny(przewodnik), żółty(ogon), czerwony(głowa)

2.4 Zapis planszy do pliku

- Przycisk ”Save” prowadzi do menu zapisu, gdzie mamy możliwość wyboru nazwy i lokalizacji pliku



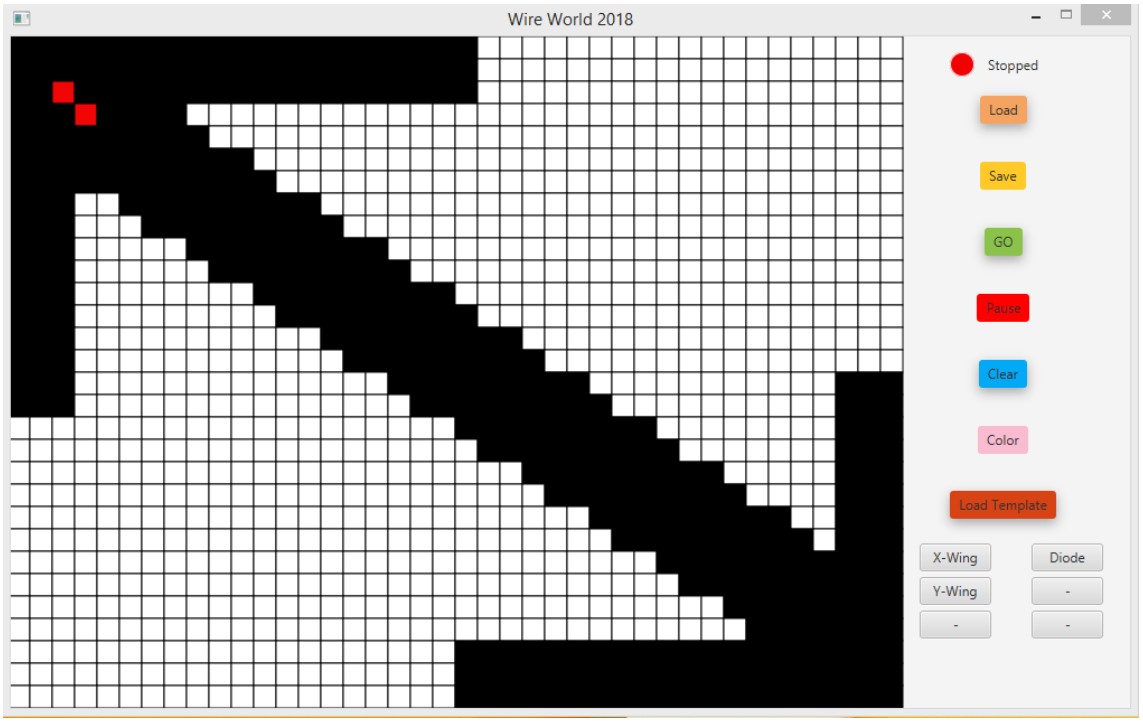
2.5 Wczytanie planszy z pliku



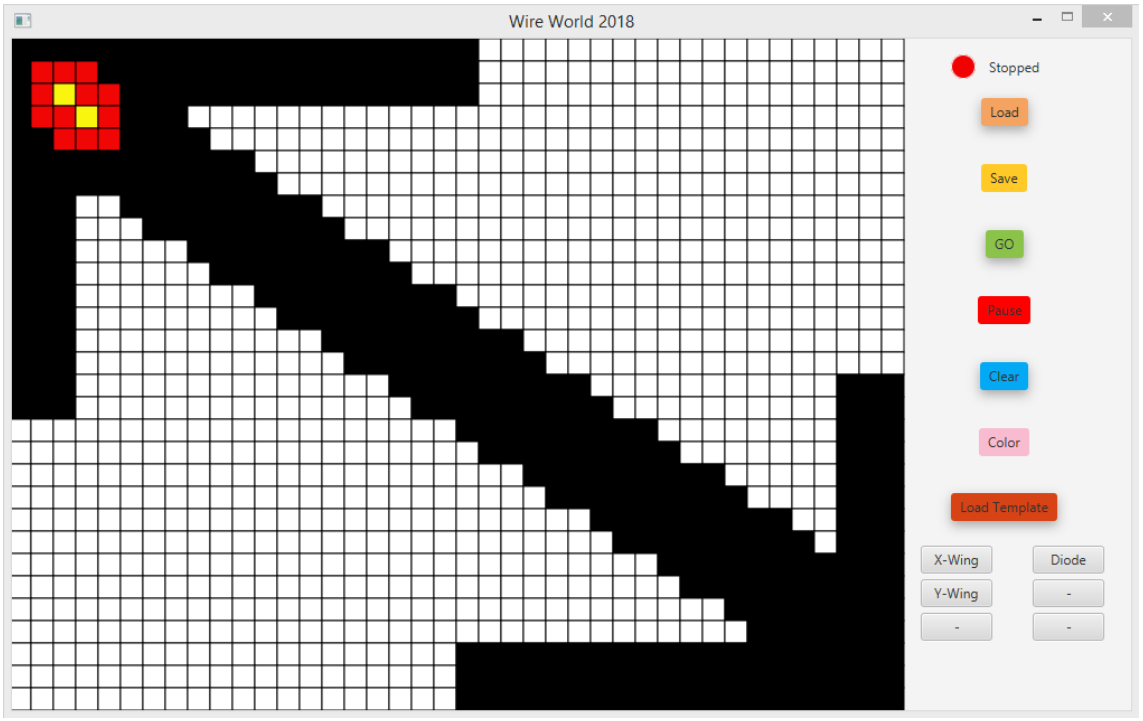
- Przycisk ’Load” prowadzi do menu wyboru plików, gdzie mamy możliwość odnalezienia nazwy i lokalizacji pliku
- Plik zostaje wczytany i od razu nałożony na plansze

2.6 Animacja

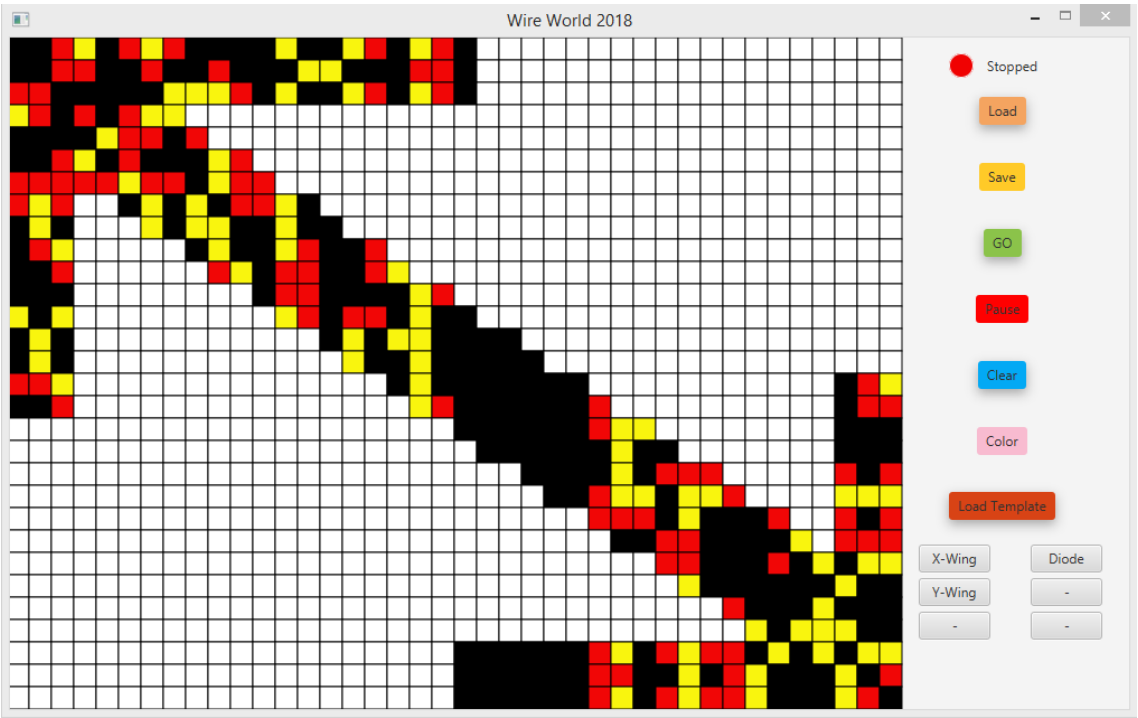
- Klatka 1 przykładowej konfiguracji planszy



- Klatka 2 przykładowej konfiguracji planszy



- Klatka n-ta przykładowej konfiguracji planszy



- Powyższe zrzuty ekranu z pracy aplikacji ukazują pracę generatora nowych klatek wspieranego przez interfejs graficzny.
- Powyższy przykład ukazuje również jaki jest domyślny sposób analizy planszy, czyli analizowanie planszy w trybie „bez ramkowym”, czyli komórki na granicach planszy wchodzą w interakcję z tymi po „drugiej stronie ekranu”.
- Zrzuty ekranu zostały wykonane przy domyślnym motywie kolorów reprezentujących stany komórek.

---

3 Opis implementacji

3.1 Implementacja modułu „Board”

1. Wprowadzono klasę Cell rozszerzającą Rectangle, aby móc wyodrębnić w niej metodę zmieniającą kolor, a także pola współrzędnych na planszy danej komórki oraz pole "poprzedniego koloru". Porządkuje to kod i ułatwia korzystanie z klasy Rectangle do naszych celów.
2. Metoda makeBoard otrzymuje dodatkowo jako argumenty wymiary obiektu Pane, na którym będzie tworzona plansza, ponieważ metody inicjujące GUI potrzebowały tej informacji
3. Dodano szereg funkcji pozwalających na kontrolę planszy:
   - repaintBoard - nadaje planszy obecne kolory i ustawia "poprzednie kolory"komórek na obecne
   - setBoardColor - nadaje całej planszy jeden kolor
   - repaintBoardOnPrevious - maluje komórki planszy na jej poprzednie kolory
   - setColorMode - ustawia tryb planszy na zwykłe kolorowanie planszy poprzez "kliknięcia"i "przeciągnięcia"
   - setCurrentBoardMode - nadaje wartość obecnego trybu planszy

- setInsensitiveMode - ustawia tryb planszy, w którym jest ona nieczuła na edycje
- setTemplateInsertionMode - ustawia tryb planszy na wprowadzanie wzorów

4. W klasie Template zdefiniowano metody wstawiania wzoru na plansze, we wszystkie cztery możliwe kierunki

3.2 Zmiany w implementacji modułu „GUI"

1. Do menu wyboru kolorów dodano przycisk powrotu do domyślnych kolorów, a także przycisk ustawienia obecnych kolorów na przyciskach wyboru koloru, co ułatwia zarządzanie kolorami planszy.

2. Do kontrolerów dodano pole Colors , aby przekazywać informacje o obecnych kolorach planszy

3. MainScreenController otrzymał pole BoardMaker, aby przekazywać mu sterowanie naszą planszą

4. MainScreenControlle otrzymał szereg metod kontrolujących elementy tej sceny:

   - disableNonTemplateButtons - blokuje dostęp do przycisków nie służących do wstawiania wzorów
   - disableTemplateButtons - blokuje dostęp do przycisków służących do wstawiania wzorów
   - isAnimationRunningSignal - pozwala na włączenie lub wyłączenie obiektu pokazującego czy animacja obecnie działa
   - manageTemplateInsertion - porządkuje działania w metodach pod przyciskami do wstawiania wzorów

3.3 Implementacja pakietu „Generation"

3.3.1 Zmiany w klasie GenerationsHandler

1. Nazwa obiektu zmieniona na „GeneratorHandler", aby lepiej wyrazić zastosowanie klasy.

2. Dodano nowe pole „running" typu boolean na potrzeby cechy wielowątkowości programu

3. Dodano dwie metody set i get dla pola „mode"

4. Dodano nowe pole „generator" typu Generation.CellularAutomation

5. Dodano funkcję „terminate", która wstrzymuje pracę generatora ostatecznie, co pozwala na przerwanie pracy wątku generatora

3.3.2 Dodano klasę StatusIndicators

1. Klasa ta zawiera nazwy stałych (String) i ich wartości (int), określające stany w których mogą się znajdować komórki planszy. tj.:

   (a) HEAD = 3
   (b) TAIL = 2
   (c) CONDUCTOR = 1
   (d) EMPTY = 0

2. Klasa zapewnia lepszą czytelność kodu dla innych klas z pakietu

3.3.3 Zmiany w klasie CellularAutomation

1. Dodano pole „adapter" typu Generation.BoardAdapter ułatwiający i standaryzujący dostęp do pól analizowanej planszy
2. Zrezygnowano z pól „headColor", „tailColor", „conductorColor" oraz „emptyFieldColor", które nie znalazły zastosowania w tej klasie
3. Dodano metodę „updateBoard", która na podstawie informacji zawartych w polu „tmpMatrix" uaktualnia stany komórek na planszy
4. Dodano metodę „setBorderss", która służy rolę przekaźnika instrukcji do obiektu „Neighbours"

3.3.4 Dodano klasę BoardAdapter

1. Obiekt ten zawiera pola „list" (ArrayList<Cell>), „height" (int) i „width" (int), które opisują oraz zawierają informacje o planszy komórek
2. Klasa ta zapełnia łatwiejszy dostęp do komórek planszy
3. Obecność klasy zwalnia pozostałe obiekty z wiedzy w jaki sposób komórki są przechowywane w pamięci

3.3.5 Zmiany w klasie Rules

1. W instrukcjach warunkowych oraz przy wyrażaniu wartości zwracanych wykorzystano stałe z obiektu Generation.StatusIndicators, co przełożyło się na lepsza czytelność kodu oraz jego łatwiejsze zrozumienie.
2. Brak zmian w strukturze oraz logice pracy obiektu

3.3.6 Zmiany w klasie Neighbours

1. Zrezygnowano z pola „rectangleMap" typu HashMap<String, javafx.scene.shape.Rectangle> na rzecz pola „adapter" Generation.BoardAdapter
2. Dodano pole „adapter" typu Generation.BoardAdapter ułatwiający i standaryzujący dostęp do pól analizowanej planszy
3. W wyniku analizy pracy programu dostrzeżono wspólne zachowanie dwóch funkcji „neighbours" i „borderlessNeighbours". W wyniku czego została wyodrębniona nowa metoda „analyzeCell", która sprawdza czy komórka o podanym indeksie znajduje się na planszy oraz w przypadku, gdy pytana komórka jest w stanie „3" - zwiększa licznik „heads"

3.4 Implementacja pakietu „IO"

3.4.1 Zmiany w klasie IO

1. Zmieniono nazwę metody „fillBoard" na „updateBoard", aby lepiej oddać znaczenie działania funkcji
2. Dodano trzy funkcje konwertujące sposób zapisu danych: „fileToIntMatrix", „boardToRGBAMatrix" oraz „intMatrixToArrayList"
3. Dodano funkcję „updateRaster" na potrzeby funkcji „saveBoard"
4. Zrezygnowano z funkcji określających kolor na rzecz czytelności oraz muskularności kodu. Wykorzystano funkcje z klasy IO.ColorPicker.

5. Dodano funkcję „readTemplate” dla wsparcia funkcji wczytywania gotowych wzorów z plików.

3.4.2 Dodano klasię ColorPicker

1. Klasa została utworzona jako wsparcie dla klasy IO.IO. Jej zadaniem jest określenie koloru na podstawie otrzymanych danych.

---

4 Opcje rozbudowy programu

1. Pakiet Generation został zaprojektowany do pracy z planszą o dowolnych rozmiarach, co oznacza, że możliwa jest stosunkowa szybka i prosta modyfikacja programu do pracy na różnych wymiarach planszy.

2. Pakiet Generation wspiera funkcję przełączania trybu analizy planszy pomiędzy trybami „z granicami” oraz „bez granic”.

3. Pakiet Generation wspiera opcję ustawienia szybkości generowania kolejnych klatek ukazujących stany komórek na planszy.