WireWorld

Aspras Rafał i Szymański Karol

04.06.2017

1 Temat i jego analiza

Program WireWorld jest automatem komórkowym i działa na zasadzie WireWorld'a Briana Silvermana. Jego celem jest przeprowadzenie symulacji elementów elektronicznych operujących na wartościach bitowych. To jest na przykład sprawdzić ja wygląda przepływ elektronu przez obwód.

2 Specyfikacja użytkownika

Program uruchamia się poprzez dwukrotne kliknięcie myszą na plik project.jar (Project/dist/project.jar). Intuicyjny interfejs aplikacji pozwoli użytkownikowi na łatwe i przyjemne korzystanie z niej. Program daje użytkownikowi sporo możliwości takich jak:

1. Wybrać rozmiar planszy (od 16x16 do 64x64)
2. Wczytać utworzoną już wcześniej planszę
3. Zapisać akutalną planszę
4. Wstawić gotowy element taki jak dioda czy bramka logiczna

Obwód oczywiście można zbudować także używając własnej inwencji poprzez korzystanie z myszy i opcje tu wyglądają następująco:

1. LPM - wstawia przewodnik / bramkę logiczną
2. PPM - usuwa element / obraca bramkę logiczną (w modzie sterowania bramką logiczną)
3. Rolka (ŚPM) - wstawia kolejno: głowę elektronu, ogon elektronu
4. ESC - będąc w modzie sterowania bramką logiczną, powoduje wyjście z niego

Przewodnik po kolorystyce elementów:

1. Niebieski - głowa elektronu

2. Czerwony - ogon elektronu
3. Żółty - przewodnik
4. Ciemno szary - pusty

Jak we wszystkich automatach komórkowych upływ czasu przedstawiony jest w postaci dyskretnych kroków czasowych, czyli generacji. Komórka pusta (w stanie zero) na zawsze pozostaje w swoim stanie; inne komórki zachowują się w następujący sposób:

1. Głowa elektronu → ogon elektronu
2. Ogon elektronu → przewodnik
3. Przewodnik → głowa elektronu, ale tylko wtedy, gdy dokładnie 1 lub 2 komórki sąsiadujące są głowami elektronu

3 Specyfikacja wewnętrzna

Formaty plików
Program korzysta z plików w formacie: .java, .w . Piki *.java są plikami wykonującymi i na nich opiera się działanie całego programu. Pliki *.w stanowią pliki wczytywania/zapisywania poszczególnych generacji.

Struktury danych
public Coords(int x,int y) - przechowuje współrzędne x, y elementu.
public CoordsState(Coords coords,State state) - przechowuje współrzędne x, y elementu i stan w którym się znajduje.
public Map(SizeofMap sizeMap) - przechowuje rozmiar mapy

Diagram klas
Hierachia klas:

- java.lang.Object
  - Box.Box (implements java.io.Serializable)
  - Gates.CleanGate
  - java.awt.Component (implements java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable)
    - java.awt.Container
      - javax.swing.JComponent (implements java.io.Serializable)
        - javax.swing.JLabel (implements javax.accessibility.Accessible, javax.swing.SwingConstants)
          - GUI.BoxColour
        - javax.swing.JPanel (implements javax.accessibility.Accessible)

        - GUI.Map (implements java.util.Observer)
    - java.awt.Window (implements javax.accessibility.Accessible)
        - java.awt.Dialog
            - javax.swing.JDialog (implements javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants)
                - GUI.CustomWindow
- Box.Coords (implements java.io.Serializable)
- Box.CoordsState
- File.FileExporter
- File.FileImporter
- Gates.Gate
    - Gates.Diode1 (implements Gates.Gates)
    - Gates.Diode1 (implements Gates.Gates)
    - Gates.GateAN (implements Gates.Gates)
    - Gates.GateOR (implements Gates.Gates)
- java.awt.event.KeyAdapter (implements java.awt.event.KeyListener)
    - Listeners.KeyListener
- java.awt.event.MouseAdapter (implements java.awt.event.MouseListener, java.awt.event.MouseMotionListener, java.awt.event.MouseWheelListener)
    - Listeners.MouseListener
- java.util.Observable
    - Simulation.NextGeneration (implements java.util.Observer)
    - Simulation.NextTick
    - Simulation.SimulationTick
- GUI.SizeofMap (implements java.io.Serializable)
- Box.State (implements java.io.Serializable)
- java.lang.Thread (implements java.lang.Runnable)
    - Simulation.Simulation