UserInterface.java

```
1   import javax.swing.*;
2
3   import org.tigam.railcab.algoritme.ReisManager;
4
5   import org.tigam.railcab.algoritme.Simulatie;
6
7   import java.awt.*;
8
9   import java.awt.event.*;
10
11  import java.applet.*;
12
13  import java.io.BufferedReader;
14  import java.io.File;
15
16  import java.io.BufferedWriter;
17
18  import java.io.FileNotFoundException;
19
20  import java.io.FileReader;
21
22  import java.io.FileWriter;
23
24  import java.io.IOException;
25
26  import java.io.PrintStream;
27
28  import java.util.EventObject;
29
30  public class UserInterface extends JFrame {
31
32      private JPanel simulatie;
33
34      private Paneel paneel;
35
36      private MenuItem momentOpslaan, momentLaden, baanInstelling, hulp,
37              bestandSluiten, handleiding, about;
38
39      private Controller controller;
40
```

```
41          private ReisManager reisManager;
42
43          private Simulatie simulatie_;
44
45          private static boolean isGestart = false;
46
47          private static boolean isGepauzeerd = false;
48
49          private JFrame frame;
50
51          public UserInterface() {
52
53              paneel = new Paneel();
54
55              simulatie = new Paneel();
56
57              setContentPane(simulatie);
58
59              controller = new Controller();
60
61              frame = new JFrame();
62
63              simulatie_ = Simulatie.getInstantie();
64
65              reisManager = simulatie_.getReisManager();
66
67              MenuBar menu = new MenuBar();
68
69              setMenuBar(menu);
70
71              /**
72               * maak de menu's aan
73               */
74
75              Menu bestand = new Menu("Bestand");
76
77              Menu opties = new Menu("Opties");
78
79              Menu help = new Menu("Help");
80
81              /**
82               * voeg items toe aan menu's
```

```
83              */
84
85             momentOpslaan = new MenuItem("Momentopname Opslaan");
86
87             baanInstelling = new MenuItem("Baan instellingen");
88
89             bestandSluiten = new MenuItem("Afsluiten ALT + F4");
90
91             handleiding = new MenuItem("Handleiding");
92
93             about = new MenuItem("About");
94
95             hulp = new MenuItem("Help");
96
97             /**
98              * Hier worden de inwendigge klasses aangeroepen als er een aanroep naar
99              * die klasses zijn gedaan
100             */
101            about.addActionListener(new About());
102
103            handleiding.addActionListener(new Handleiding());
104
105            baanInstelling.addActionListener(new Instelling());
106
107            bestandSluiten.addActionListener(new Sluiten());
108
109            momentOpslaan.addActionListener(new MomentOpnameOpslaan());
110
111            /**
112             * voeg menu toe aan balk
113             */
114
115            menu.add(bestand);
116
117            menu.add(opties);
118
119            menu.add(help);
120
121            /**
122             * voeg menuitems toe aan 'bestand'
123             */
124
```

```
125             bestand.add(momentOpslaan);
126 
127             bestand.add(bestandSluiten);
128 
129             /**
130              * voeg menuitems toe aan 'opties'
131              */
132 
133             opties.add(baanInstelling);
134 
135             /**
136              * voeg menuitems toe aan 'help'
137              */
138 
139             help.add(handleiding);
140 
141             help.add(about);
142 
143         }
144 
145         /**
146          * Deze methode houd de status bij van het simulatie. of het gestart is of
147          * niet gestart is d..m.v een true en false
148          */
149         public static boolean isGestart(boolean test) {
150 
151             isGestart = test;
152 
153             return isGestart;
154 
155         }
156 
157         /**
158          * Deze methode houd de status bij van het simulatie. of het gepuazeerd is
159          * of niet gepauzeerd is d..m.v een true en false
160          */
161         public static boolean isGepauzeerd(boolean test) {
162 
163             isGepauzeerd = test;
164 
165             return isGepauzeerd;
166 
```

```
167         }
168 
169         /**
170          * inwendige klasse voor het openen van de instellingen klasse in een frame.
171          */
172 
173         class Instelling implements ActionListener {
174 
175             public void actionPerformed(ActionEvent e) {
176 
177                 new InstellingScherm();
178 
179             }
180 
181         }
182 
183         /**
184          * inwendige klasse voor het sluiten van de simulatie.
185          */
186         class Sluiten implements ActionListener {
187 
188             public void actionPerformed(ActionEvent e) {
189 
190                 dispose();
191 
192             }
193 
194         }
195 
196         /**
197          * inwendige klasse voor het opslaan van een momentopname Die heeft tevens
198          * een voorwaarde. Het mag alleen gedaan worden als de simulatie draait
199          */
200         class MomentOpnameOpslaan implements ActionListener {
201 
202             public void actionPerformed(ActionEvent e) {
203 
204                 if (isGestart == true
205                         || reisManager.getStatus() == ReisManager.START) {
206 
207                     if (isGepauzeerd == false) {
208 
```

```
209                     System.out.println("voert hij dit uit " + isGepauzeerd);
210
211                     controller.momentOpnameOpslaan();
212                     JOptionPane.showMessageDialog(frame, "Bestand opgeslagen");
213                 } else
214
215                 {
216
217                     JOptionPane.showMessageDialog(frame,
218                             "De Simulatie is gepauzeerd", "Fout",
219
220                             JOptionPane.ERROR_MESSAGE);
221
222                 }
223
224             } else {
225
226                 JOptionPane.showMessageDialog(frame,
227                         "De Simulatie is nog niet gestart", "Fout",
228
229                         JOptionPane.ERROR_MESSAGE);
230
231             }
232
233         }
234
235     }
236
237     /**
238      * inwendige klasse voor het openen van de about frame
239      */
240     class About implements ActionListener {
241         public void actionPerformed(ActionEvent e) {
242             new About();
243         }
244     }
245
246     /**
247      * inwendige klasse voor openen van de handleiding pdf
248      */
249     class Handleiding implements ActionListener {
250         public void actionPerformed(ActionEvent e) {
```

```
251
252             boolean exists = (new File("src/handleiding.pdf")).exists();
253
254             if (exists) {
255                 try {
256                     Runtime.getRuntime().exec(
257                             "rundll32 url.dll,FileProtocolHandler "
258                                     + "src/handleiding.pdf");
259                 } catch (IOException e1) {
260                     // TODO Auto-generated catch block
261                     e1.printStackTrace();
262                 }
263             } else {
264
265                 JOptionPane.showMessageDialog(frame,
266                         "Handleiding niet gevonden", "Fout",
267                         JOptionPane.ERROR_MESSAGE);
268             }
269         }
270     }
271
272     public static void main(String args[]) {
273
274         JFrame frame = new UserInterface();
275
276         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
277
278         frame.setBounds(0, 0, 1024, 768);
279
280         frame.setVisible(true);
281
282     }
283
284 }
285
```

TekenSpoor.java

```
1   import java.awt.BorderLayout;
2
3   import java.awt.Container;
4   import java.awt.Frame;
5   import java.awt.Graphics;
6   import java.awt.Image;
7   import java.awt.MediaTracker;
8   import java.awt.TextField;
9   import java.awt.Toolkit;
10  import java.awt.event.ActionEvent;
11  import java.awt.event.WindowAdapter;
12  import java.awt.event.WindowEvent;
13  import java.awt.*;
14  import java.awt.event.*;
15  import java.net.URL;
16
17  import javax.swing.*;
18
19  import javax.swing.ImageIcon;
20  import javax.swing.JButton;
21  import javax.swing.JFrame;
22  import javax.swing.JLabel;
23  import javax.swing.JOptionPane;
24  import javax.swing.JPanel;
25  import javax.swing.JTextField;
26
27  public class TekenSpoor {
28      private JFrame frame;
29      private JPanel spoor;
30      private JTextField aStation, aTaxi;
31      private JButton submit;
32      private ImageIcon pSpoor;
33      private int x, y = 0;
34      private Paneel paneel;
35      private JLabel notification, taxi_not;
36      private Controller controller;
37
38      public TekenSpoor(){
39      }
40
41      public TekenSpoor(Paneel paneel){
42          this.paneel = paneel;
43
44          pSpoor = new ImageIcon("src/spoor.gif");
45          frame = new JFrame("Spoor Instellingen");
46          spoor = new JPanel()
47
48          {
49              protected void paintComponent(Graphics g)
50              {
51
52                  g.drawImage(pSpoor.getImage(), 0, 0, null);
```

```
53
54                  super.paintComponent(g);
55              }
56          };
57
58          notification = new JLabel();
59          notification.setSize(300,25);
60          notification.setLocation(20, 200);
61          //controller = new Controller();
62
63          taxi_not = new JLabel();
64          taxi_not.setSize(300,25);
65          taxi_not.setLocation(20, 220);
66
67          aStation = new JTextField();
68          aStation.setSize(120,25);
69          aStation.setLocation(20, 90);
70
71          aTaxi = new JTextField();
72          aTaxi.setSize(120,25);
73          aTaxi.setLocation(20,180);
74
75          submit = new JButton("OK");
76          submit.setSize(53,30);
77          submit.setLocation(110,230);
78          submit.addActionListener(new Submit());
79
80
81          //voeg alles toe
82          frame.add( taxi_not );
83          frame.add( aStation );
84          frame.add( aTaxi );
85          frame.add( submit );
86          frame.add( notification );
87
88          //tekent het scherm
89          spoor.setOpaque( false );
90          frame.getContentPane().add(spoor, BorderLayout.CENTER);
91          frame.setSize( 300, 300);
92          frame.setVisible( true );
93
94      }
95
96      class Submit implements ActionListener {
97          public void actionPerformed(ActionEvent e) {
98              //probeer de waarde uit de tekstvakken te halen...
99              try{
100                 //haal de waardes op uit het tekstvak voor het station
101                 String aStat = aStation.getText();
102                 //...en zet het in een variabele
103                 int aantal = Integer.parseInt(aStat);
104
105                 //haal de waardes op uit het tekstvak voor de taxi
106                 String aTax = aTaxi.getText();
107                 //...en zet het in een variabele
```

```
108                 int aantTaxi = Integer.parseInt(aTax);
109
110                 //kijk nu of het wel geldige waardes zijn
111                 if(aantTaxi > 1 && aantTaxi <= 8 && aantal >1 && aantal <=8){
112                     //en kies dan een situatie aan de hand van de ingevoerde waarde
113                     //vervolgens stuurt hij een opdracht naar de paneel klasse
114                     switch (aantal) {
115                     case 0: break;
116                     case 1: break;
117                     case 2: paneel.tweeStations(); break;
118                     case 3: paneel.drieStations(); break;
119                     case 4: paneel.vierStations(); break;
120                     case 5: paneel.vijfStations(); break;
121                     case 6: paneel.zesStations(); break;
122                     case 7: paneel.zevenStations(); break;
123                     case 8: paneel.achtStations(); break;
124                     default: break;
125                     }
126                 }
127                 //voor de taxi's
128                 //hier staat een extra opdracht, voor het doorgeven aan het algoritme
129                 if(aantTaxi > 1 && aantTaxi <= 8 && aantal >1 && aantal <=8){
130                     paneel.setTaxi(aantTaxi);
131
132                 }
133                 //en kies dan een situatie aan de hand van de ingevoerde waarde
134                 //vervolgens stuurt hij een opdracht naar de paneel klasse
135                 switch (aantTaxi) {
136                 case 0: break;
137                 case 1: break;
138                 case 2: paneel.tweeTaxi(); break;
139                 case 3: paneel.drieTaxi(); break;
140                 case 4: paneel.vierTaxi(); break;
141                 case 5: paneel.vijfTaxi(); break;
142                 case 6: paneel.zesTaxi(); break;
143                 case 7: paneel.zevenTaxi(); break;
144                 case 8: paneel.achtTaxi(); break;
145                 default: break;
146                 }
147
148                 //als alle waardes correct ingevuld zijn, sluit dan netjes het scherm af
149                 if(!( aantal <= 1 || aantTaxi < 2 || aantal > 8 || aantTaxi > 8)){
150                     frame.dispose();
151                 }
152                 //als er 1 of meerdere verkeerde waardes ingevoerd zijn, geef dan een waarschuwing
153                 if( aantal <= 1 || aantTaxi < 2 || aantal > 8 || aantTaxi > 8){
154                     JOptionPane.showMessageDialog(frame, "Niet meer dan 8 taxi's en minder dan 2.\nNiet meer dan 8 stations en minder dan 2.","Fout",
155                             JOptionPane.ERROR_MESSAGE);
156                 }
157             }
158
159             //als er helemaal geen waardes ingevuld zijn, geef ook een waarschuwing
160             catch(NumberFormatException e1){
161                 JOptionPane.showMessageDialog(frame, "Vul een geldige waarde in.","Fout",
162                         JOptionPane.ERROR_MESSAGE);
```

```
163                 }
164
165
166
167
168             }
169         }
170     }
171
```

TaxiStats.java

```
1           /*-----------------------------*/
2          /* This Program was made by:   */
3         /* Isabelle Hanraads           */
4        /* Any kind of fraude will be  */
5       /* seen as a serious crime and */
6      /* will be punished by a money */
7     /* penalty of 25,-             */
8    /*_____________________________*/
9
10   import java.awt.BorderLayout;
11   import java.awt.Frame;
12   import java.awt.Graphics;
13   import java.awt.Font;
14   import java.awt.event.ActionEvent;
15   import java.awt.event.ActionListener;
16
17   import javax.swing.*;
18
19   import org.tigam.railcab.algoritme.*;
20
21
22   public class TaxiStats extends Frame{
23       private JFrame frame;
24       private JPanel tStats;
25       private JLabel positie, r_inTaxi, wTijd, rTijd, error, message, message2, bestemming;
26       private ImageIcon bg, reiziger;
27       public String pos, Str_r_inT, Str_rT_t;
28       public int r_inT;
29       private String rT, Str_best, rNaam, naam, naamR1, naamR2, naamR3, naamR4, naamR5;
30       private JButton close;
31       private JButton reiziger1, reiziger2, reiziger3, reiziger4, reiziger5;
32       private Reis reis;
33       private Reiziger reizig;
34       private Controller controller;
35       private String[] stringArray;
36       private BaanManager baanManager;
37       private ReisManager reisManager;
38       private int ID = 0, i, y;
39
40
41
42       public TaxiStats(String oposs, int r_inT, String Str_rT_t, int ID){
43           rNaam = rNaam;
44           this.ID = ID;
45           frame = new JFrame("Taxi Statistieken");
46           bg = new ImageIcon( "src/taxi_bg.gif" );
47           reiziger = new ImageIcon( "src/reiziger.gif");
48           tStats = new JPanel(){
49               protected void paintComponent(Graphics g)
50               {
51                   g.drawImage(bg.getImage(), 0, 0, null);
52
53                   super.paintComponent(g);
54               }
55           };
56
57           baanManager = Simulatie.getInstantie().getBaanManager();
58           reisManager = Simulatie.getInstantie().getReisManager();
59
60           // convert alle integer statistieken naar Strings
61           String Str_r_inT = Integer.toString(r_inT);
62
63           //voeg de knoppen voor de reizigerstats toe
64           reiziger1 = new JButton(reiziger);
65           reiziger1.setLocation(212, 106);
66           reiziger1.setSize(15,30);
67           reiziger1.setVisible( false );
68           reiziger1.addActionListener(new Reiziger1());
69
```

```
70          reiziger2 = new JButton(reiziger);
71          reiziger2.setLocation(245, 106);
72          reiziger2.setSize(15,30);
73          reiziger2.setVisible( false );
74          reiziger2.addActionListener(new Reiziger2());
75
76          reiziger3 = new JButton(reiziger);
77          reiziger3.setLocation(228, 135);
78          reiziger3.setSize(15,30);
79          reiziger3.setVisible( false );
80          reiziger3.addActionListener(new Reiziger3());
81
82          reiziger4 = new JButton(reiziger);
83          reiziger4.setLocation(212, 165);
84          reiziger4.setSize(15,30);
85          reiziger4.setVisible( false );
86          reiziger4.addActionListener(new Reiziger4());
87
88          reiziger5 = new JButton(reiziger);
89          reiziger5.setLocation(245, 165);
90          reiziger5.setSize(15,30);
91          reiziger5.setVisible( false );
92          reiziger5.addActionListener(new Reiziger5());
93
94
95          // voeg alle labels toe!
96          positie = new JLabel(oposs);
97          positie.setLocation(27,67);
98          positie.setSize(200,100);
99          positie.setFont( new Font( "Arial", Font.BOLD, 16) );
100
101         r_inTaxi = new JLabel(Str_r_inT + " reizigers");
102         r_inTaxi.setLocation(27,115);
103         r_inTaxi.setSize(200,100);
104         r_inTaxi.setFont( new Font( "Arial", Font.BOLD, 16) );
105
106         rTijd = new JLabel (Str_rT_t + " minuten");
107         rTijd.setLocation(27,170);
108         rTijd.setSize(200,100);
109         rTijd.setFont( new Font( "Arial", Font.BOLD, 16) );
110
111         bestemming = new JLabel();
112         bestemming.setLocation(27,220);
113         bestemming.setSize(200,100);
114         bestemming.setFont( new Font( "Arial", Font.BOLD, 16) );
115
116         error= new JLabel ();
117         error.setLocation(27,270);
118         error.setSize(200,100);
119         error.setFont( new Font( "Arial", Font.BOLD, 15) );
120
121         message= new JLabel ();
122         message.setLocation(27,290);
123         message.setSize(200,100);
124         message.setFont( new Font( "Arial", Font.BOLD, 15) );
125
126         message2= new JLabel ();
127         message2.setLocation(27,310);
128         message2.setSize(200,100);
129         message2.setFont( new Font( "Arial", Font.BOLD, 15) );
130
131         close = new JButton ("Sluiten");
132         close.setLocation(175, 370);
133         close.setSize(100,25);
134         close.addActionListener( new Close());
135
136         frame.add(bestemming);
137         frame.add(message2);
138         frame.add(message);
139         frame.add(error);
140         frame.add(positie);
```

```
141         frame.add(r_inTaxi);
142         frame.add(rTijd);
143         frame.add( close );
144         frame.add( reiziger1 );
145         frame.add( reiziger2 );
146         frame.add( reiziger3 );
147         frame.add( reiziger4 );
148         frame.add( reiziger5 );
149 
150         tStats.setOpaque( false );
151         frame.getContentPane().add(tStats, BorderLayout.CENTER);
152         frame.setSize( 300, 450);
153         frame.setVisible( true );
154 
155         //kijk of het aantal reizigers in een door 'ID' gekozen taxi, niet op nul staat.
156         //set in de variabele Str_best, de bestemming uit de baanManager.
157         //zet dan de tekst voor Label 'bestemming' op de bovenstaande variabele
158         if(!(baanManager.getTaxi(ID).getAantalReizigers() == 0)){
159             Str_best = baanManager.getTaxi(ID).getReis().getReisDetails().getBestemming().getNaam();
160             bestemming.setText(Str_best);
161             }
162 
163         //zet het aantal reizigerbuttons op variabele r_inT, wat doorgegeven wordt via de constructor
164         switch(r_inT){
165         case 0: break;
166         case 1: reiziger1.setVisible(true); break;
167         case 2: reiziger1.setVisible(true); reiziger2.setVisible(true); break;
168         case 3: reiziger1.setVisible(true);reiziger2.setVisible(true);reiziger3.setVisible(true); break;
169         case 4: reiziger1.setVisible(true);reiziger2.setVisible(true);reiziger3.setVisible(true);
170                 reiziger4.setVisible(true); break;
171         case 5: reiziger1.setVisible(true);reiziger2.setVisible(true);reiziger3.setVisible(true);
172                 reiziger4.setVisible(true); reiziger5.setVisible(true); break;
173         default: break;
174         }
175 
176         //kijk voor iedere waarde van 'i' (totdat deze gelijk is het aantal reizigers in een taxi
177         //gekozen met 'ID', zolang deze kleiner is, tel dan 1 bij 'i' op.
178 
179         //zet voor iedere keer dat 'i' kleiner is dan het aantal reizigers(gehaald uit de baanmanager)
180         //de variabele String 'rNaam' op de naam van de reiziger (gehaald uit een array, uit de BaanManager)
181         for (i=0; i < baanManager.getTaxi(ID).getAantalReizigers(); i++){
182             System.out.println("Naam: " + baanManager.getTaxi(ID).getReizigerNamen()[i]);
183             System.out.println("index: " +i);
184             rNaam = baanManager.getTaxi(ID).getReizigerNamen()[i];
185             //kijk of i voldoet aan de voorwaarde
186             //en zet dan de variabele 'rNaam' in de variabele per reiziger 'naamR1, naamR2, etc'
187             if( i == 0){
188                 naamR1 = rNaam;
189             }
190 
191             if( i == 1){
192                 naamR2 = rNaam;
193             }
194 
195             if( i == 2){
196                 naamR3 = rNaam;
197             }
198 
199             if( i == 3){
200                 naamR4 = rNaam;
201             }
202 
203             if( i == 4){
204                 naamR5 = rNaam;
205             }
206         }
207     }
208     //kijk of het aantal seconde niet boven de 60 uit komt...
209     public int secondeCheck(int sec){
210         for(int i = 0; sec > 60; i++){
211             sec = sec - 60;
```

```
212         }
213         return sec;
214     }
215
216     //interne klasse voor reiziger 1, voor de andere reizigers is het precies hetzelfde
217     class Reiziger1 implements ActionListener{
218         public void actionPerformed(ActionEvent e){
219
220             //kijk of de taxi reiziger met de naam 'naamR1' bevat.
221             if( baanManager.getTaxi(ID).bevatReiziger(naamR1)){
222
223             String Str_wT_r;
224             String Str_rT_r;
225
226             //haal de wachttijd op station uit de baanManager.
227             int minuut = (int) (baanManager.getTaxi(ID).getReiziger(naamR1).getWachtTijd()/1000/60);
228             int seconde = (int) (baanManager.getTaxi(ID).getReiziger(naamR1).getWachtTijd()/1000);
229             Str_wT_r = minuut +"."+secondeCheck(seconde);
230
231             //haal de reistijd uit de baanManager.
232             int min = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000/60);
233             int sec = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000);
234             Str_rT_r = min + "."+secondeCheck(sec);
235             //maak een nieuwe instantie met de zojuist gekregen waardes
236             new ReizerStat(naamR1,Str_wT_r, Str_rT_r);
237
238             }
239             //zodra de reiziger is uitgestapt, geef dan een waarschuwing.
240             else{
241                 JOptionPane.showMessageDialog(frame, naamR1 + "is al uitgestapt!\nOpen het scherm opnieuw,\nvoor de juiste gegevens.","Fout",
242                         JOptionPane.ERROR_MESSAGE);
243             }
244         }
245     }
246     class Reiziger2 implements ActionListener{
247         public void actionPerformed(ActionEvent e){
248
249             if( baanManager.getTaxi(ID).bevatReiziger(naamR2)){
250             String Str_wT_r ;
251             String Str_rT_r;
252
253             //wachttijd op station
254             int minuut = (int) (baanManager.getTaxi(ID).getReiziger(naamR2).getWachtTijd()/1000/60);
255             int seconde = (int) (baanManager.getTaxi(ID).getReiziger(naamR2).getWachtTijd()/1000);
256             Str_wT_r = minuut +"."+secondeCheck(seconde);
257
258             //reistijd + wachttijd op station
259             int min = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000/60);
260             int sec = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000);
261             Str_rT_r = min + "."+secondeCheck(sec);
262             new ReizerStat(naamR2, Str_wT_r, Str_rT_r);
263             }
264             else{
265                 JOptionPane.showMessageDialog(frame, naamR2 + "is al uitgestapt!\nOpen het scherm opnieuw,\nvoor de juiste gegevens.","Fout",
266                         JOptionPane.ERROR_MESSAGE);
267             }
268         }
269     }
270     class Reiziger3 implements ActionListener{
271         public void actionPerformed(ActionEvent e){
272             if( baanManager.getTaxi(ID).bevatReiziger(naamR3)){
273
274             String Str_wT_r;
275             String Str_rT_r;
276
277             //wachttijd op station
278             int minuut = (int) (baanManager.getTaxi(ID).getReiziger(naamR3).getWachtTijd()/1000/60);
279             int seconde = (int) (baanManager.getTaxi(ID).getReiziger(naamR3).getWachtTijd()/1000);
280             Str_wT_r = minuut +"."+secondeCheck(seconde);
281
282             //reistijd + wachttijd op station
```

```
283             int min = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000/60);
284             int sec = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000);
285             Str_rT_r = min + "."+secondeCheck(sec);
286                 new ReizerStat(naamR3, Str_wT_r, Str_rT_r);
287             }
288             else{
289                 JOptionPane.showMessageDialog(frame, naamR3 + "is al uitgestapt!\nOpen het scherm opnieuw,\nvoor de juiste gegevens.","Fout",
290                         JOptionPane.ERROR_MESSAGE);
291             }
292         }
293     }
294     class Reiziger4 implements ActionListener{
295         public void actionPerformed(ActionEvent e){
296             if( baanManager.getTaxi(ID).bevatReiziger(naamR4)){
297 
298             String Str_wT_r;
299             String Str_rT_r;
300 
301             //wachttijd op station
302             int minuut = (int) (baanManager.getTaxi(ID).getReiziger(naamR4).getWachtTijd()/1000/60);
303             int seconde = (int) (baanManager.getTaxi(ID).getReiziger(naamR4).getWachtTijd()/1000);
304             Str_wT_r = minuut +"."+secondeCheck(seconde);
305 
306             //reistijd + wachttijd op station
307             int min = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000/60);
308             int sec = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000);
309             Str_rT_r = min + "."+secondeCheck(sec);
310 
311             new ReizerStat(naamR4, Str_wT_r, Str_rT_r);
312             }
313             else{
314                 JOptionPane.showMessageDialog(frame, naamR4 + "is al uitgestapt!\nOpen het scherm opnieuw,\nvoor de juiste gegevens.","Fout",
315                         JOptionPane.ERROR_MESSAGE);
316             }
317 
318         }
319     }
320     class Reiziger5 implements ActionListener{
321         public void actionPerformed(ActionEvent e){
322             if( baanManager.getTaxi(ID).bevatReiziger(naamR5)){
323 
324             String Str_wT_r;
325             String Str_rT_r;
326 
327             //wachttijd op station
328             int minuut = (int) (baanManager.getTaxi(ID).getReiziger(naamR5).getWachtTijd()/1000/60);
329             int seconde = (int) (baanManager.getTaxi(ID).getReiziger(naamR5).getWachtTijd()/1000);
330             Str_wT_r = minuut +"."+secondeCheck(seconde);
331 
332             //reistijd + wachttijd op station
333             int min = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000/60);
334             int sec = (int) (baanManager.getTaxi(ID).getReis().getReistijdBestemming()/1000);
335             Str_rT_r = min + "."+secondeCheck(sec);
336                 new ReizerStat(naamR5, Str_wT_r, Str_rT_r);
337             }
338             else{
339                 JOptionPane.showMessageDialog(frame, naamR5 + "is al uitgestapt!\nOpen het scherm opnieuw,\nvoor de juiste gegevens.","Fout",
340                         JOptionPane.ERROR_MESSAGE);
341             }
342 
343         }
344     }
345 
346     class Close implements ActionListener{
347         public void actionPerformed(ActionEvent e){
348             frame.dispose();
349         }
350 
351     }
352 }
353 
```

StationStats.java

```
1
2
3     import java.awt.BorderLayout;
4     import java.awt.Component;
5
6     import java.awt.Frame;
7
8     import java.awt.Graphics;
9
10    import java.awt.Font;
11
12    import java.awt.event.ActionEvent;
13
14    import java.awt.event.ActionListener;
15
16    import java.util.ArrayList;
17
18    import javax.swing.*;
19
20    import org.tigam.railcab.algoritme.BaanManager;
21
22    import org.tigam.railcab.algoritme.Simulatie;
23
24    /**
25     * @author dareal
26     *  Deze klasse toont de station statustieken. En wordt aangeroepen zorda je als gebruiker
27     * op een station knop drukt.
28     */
29    public class StationStats {
30
31        private JFrame frame;
32
33        private JPanel sStats;
34
35        private JLabel naam, r_inStation, t_inStation;
36
37        public int r_inStat, t_inStat;
38
39        public String nStation,station;
40
41        private ImageIcon bg;
42
43        private JButton close;
44
45        private JComboBox reizigerCombo;
46
47        private ArrayListComboBoxModel reizigerID;
48
49        private ArrayList<String> reizigerNaam;
50
51        private Simulatie simulatie;
52
53        private BaanManager baanManager;
54
55        private Controller controller;
56        private String [] spatie;
57        /**
58         * Om de klasse aan te roepen wordt de constoctor gevult met de naam van de station en
59         * de aantal reiziger op het station en de aantal taxi op het station
60         **/
61        public StationStats(String nStation, int r_inStat, int t_inStat){
62
63            frame = new JFrame("Station Statistiek");
64
65            bg = new ImageIcon( "src/station_stats.gif" );
66
67            sStats = new JPanel(){
68
69                protected void paintComponent(Graphics g)
```

```
70
71              {
72
73                  g.drawImage(bg.getImage(), 0, 0, null);
74
75                  super.paintComponent(g);
76
77              }
78
79          };
80
81          simulatie = Simulatie.getInstantie();
82
83          baanManager = simulatie.getBaanManager();
84
85          controller = new Controller();
86          /*
87           * Hieronder wordt een label aangemaakt met de aantal reizigers op het station
88           * */
89          String Str_rS = Integer.toString(r_inStat);
90
91          String Str_tS = Integer.toString(t_inStat);
92          /*Hier wordt een label gemaakt met de station naam
93           * */
94          naam = new JLabel(nStation);
95
96          naam.setLocation(20,60);
97
98          naam.setSize(200,100);
99
100         naam.setFont( new Font( "Arial", Font.BOLD, 16) );
101         //label wordt aangemaakt
102         r_inStation = new JLabel (Str_rS +" reizigers");
103
104         r_inStation.setLocation(20,125);
105
106         r_inStation.setSize(200,100);
107
108         r_inStation.setFont( new Font( "Arial", Font.BOLD, 16) );
109         // Label wordt aangemaakt met de aantal taxis
110         t_inStation = new JLabel (Str_tS + " taxi's");
111
112         t_inStation.setLocation(20,200);
113
114         t_inStation.setSize(200,100);
115
116         t_inStation.setFont( new Font( "Arial", Font.BOLD, 16) );
117         /**
118          * Hieronder wordt een combobox aangemaakt en gevult met de aantal rezeizigers
119          * op dat station.
120          * */
121         reizigerCombo = new JComboBox();
122         reizigerCombo.setLocation(140, 165);
123         reizigerCombo.setSize(100,25);
124         reizigerCombo.setModel(new DefaultComboBoxModel(baanManager.getStation(nStation).getReizigerNamen()));
125         reizigerCombo.setSelectedIndex(-1);
126
127         reizigerCombo.addActionListener( new ReizigerStat());
128
129         station = nStation;
130
131         close = new JButton ("Sluiten");
132
133         close.addActionListener (new Close());
134
135         // Alles wordt toegevoegd
136         frame.add(naam);
137
138         frame.add(r_inStation);
139
140         frame.add(t_inStation);
```

```
141
142         frame.add(reizigerCombo);
143
144         sStats.setOpaque( false );
145
146         frame.getContentPane().add(sStats, BorderLayout.CENTER);
147
148         frame.setSize( 300, 300);
149
150         frame.setVisible( true );
151
152
153     }
154     public String getStation(){
155
156         return nStation;
157
158     }
159     /**
160      * inwendige klasse voor het sluiten van het scherm
161      * */
162     class Close implements ActionListener{
163
164         public void actionPerformed(ActionEvent e){
165
166             frame.dispose();
167
168         }
169
170     }
171     /**
172      * inwendige klasse voor het aanroepen van de reiziger statustieken. Je ziet hoelang een reiziger wacht op een station
173      * Je ziet de bestemming van een reiziger
174      * Tevens verschijnt deze scherm als je op een reiziger klikt in de combobox. De getselectedItem zorgd daarvoor
175      *
176      * */
177
178     class ReizigerStat implements ActionListener{
179
180         public void actionPerformed(ActionEvent e) {
181             String reis = (String) reizigerCombo.getSelectedItem();
182
183             int ReisTijdUur = (int) (baanManager.getStation(station).getReiziger(reis).getWachtTijd()/1000/60/60);
184
185             int ReistijdSec = (int) (baanManager.getStation(station).getReiziger(reis).getWachtTijd()/1000);
186
187             int ReistijdMin = (int) baanManager.getStation(station).getReiziger(reis).getWachtTijd()/1000/60;
188
189             System.out.println(reis +
190
191                     baanManager.getStation(station).getReiziger(reis).getReisDetails().getBestemming().getNaam()
192
193                     +
194
195                     baanManager.getStation(station).getReiziger(reis).getWachtTijd());
196
197             System.out.println(reis);
198
199             System.out.println(station);
200
201             JOptionPane optionPane = new ReizigerStatistiek();
202
203             optionPane.setMessage("Naam: " + reis + "\n" + "Bestemming: " +
204
205                     baanManager.getStation(station).getReiziger(reis).getReisDetails().getBestemming().getNaam()
206
207                     + "\n"+ "Wachttijd: "+ ReisTijdUur +":"+ controller.secondeCheck(ReistijdMin) +":"+ controller.secondeCheck(ReistijdSec));
208
209             optionPane.setMessageType(JOptionPane.INFORMATION_MESSAGE);
210
211             JDialog dialog = optionPane.createDialog(null, "Reizigers Statistiek");
```

```
212
213                 dialog.setVisible(true);
214                 reizigerCombo.setSelectedIndex(-1);
215
216             }
217
218         }
219
220         class ReizigerStatistiek extends JOptionPane {
221
222             ReizigerStatistiek() {
223
224             }
225
226         }
227
228     }
229
230
231
```

ReizerStat.java

```
1   import java.awt.BorderLayout;
2   import java.awt.Frame;
3   import java.awt.Graphics;
4   import java.awt.Font;
5   import java.awt.event.ActionEvent;
6   import java.awt.event.ActionListener;
7
8   import javax.swing.*;
9
10  import javax.swing.ImageIcon;
11  import javax.swing.JButton;
12  import javax.swing.JFrame;
13  import javax.swing.JLabel;
14  import javax.swing.JPanel;
15
16  import org.tigam.railcab.algoritme.BaanManager;
17  import org.tigam.railcab.algoritme.Reis;
18  import org.tigam.railcab.algoritme.ReisManager;
19  import org.tigam.railcab.algoritme.Reiziger;
20  import org.tigam.railcab.algoritme.Simulatie;
21
22  public class ReizerStat {
23      private JFrame frame;
24      private JPanel rStats;
25      private JLabel naam, reisTijd, wachtTijd;
26      public int rTijd, wTijd, ID, minuut = 0, seconde=0;
27      public String nReiziger, Str_wT, Str_rT;
28      private ImageIcon bg;
29      private JButton close;
30
31      //krijg de waardes van TaxiStat en zet ze in variabele
32      public ReizerStat(String nReiziger, String Str_wT, String Str_rT){
33          frame = new JFrame("Reiziger Statistieken");
34          bg = new ImageIcon( "src/reiziger_stats.gif" );
35
36          this.nReiziger = nReiziger;
37          this.ID = ID;
38
39          rStats = new JPanel(){
40              protected void paintComponent(Graphics g)
```

```
41                {
42                    g.drawImage(bg.getImage(), 0, 0, null);
43
44                    super.paintComponent(g);
45                }
46            };
47
48
49            //zet in de label de naam van de reiziger (meegekregen van TaxiStat)
50            naam = new JLabel(nReiziger);
51            naam.setLocation(20,60);
52            naam.setSize(200,100);
53            naam.setFont( new Font( "Arial", Font.BOLD, 16) );
54
55            //de totale reistijd(meegekregen van TaxiStats)
56            reisTijd = new JLabel (Str_rT + " minuten");
57            reisTijd.setLocation(20,200);
58            reisTijd.setSize(200,100);
59            reisTijd.setFont( new Font( "Arial", Font.BOLD, 16) );
60
61            //wachttijd in station(meegekregen van TaxiStats)
62            wachtTijd = new JLabel (Str_wT + " minuten");
63            wachtTijd.setLocation(20,125);
64            wachtTijd.setSize(200,100);
65            wachtTijd.setFont( new Font( "Arial", Font.BOLD, 16) );
66
67            //voeg alles toe aan het frame
68        frame.add(naam);
69        frame.add(reisTijd);
70        frame.add(wachtTijd);
71
72
73        rStats.setOpaque( false );
74        frame.getContentPane().add(rStats, BorderLayout.CENTER);
75        frame.setSize( 300, 300);
76        frame.setVisible( true );
77
78        }
79
80    //inwendige klasse voor het afhandelen van het sluiten van het frame
81    class Close implements ActionListener{
82        public void actionPerformed(ActionEvent e){
```

```
83              frame.dispose();
84              }
85          }
86
87      }
88
```

Paneel.java

```
1   /*-----------------------------*/
2   /* This Program was made by:  */
3   /* Isabelle Hanraads          */
4   /* Any kind of fraude will be  */
5   /* seen as a serious crime and */
6   /* will be punished by a money */
7   /* penalty of 25,-            */
8   /*_____________________________*/
9   import java.awt.Color;
10  import java.awt.FileDialog;
11  import java.awt.Frame;
12  import java.awt.Graphics;
13  import java.awt.event.ActionEvent;
14  import java.awt.event.ActionListener;
15  import java.util.ArrayList;
16  import java.util.Observable;
17  import java.util.Observer;
18  import java.awt.Font;
19  import java.io.BufferedReader;
20  import java.io.File;
21  import java.io.FileReader;
22  import java.io.PrintStream;
23
24  import javax.swing.ImageIcon;
25  import javax.swing.JButton;
26  import javax.swing.JComboBox;
27  import javax.swing.JDialog;
28  import javax.swing.JFrame;
29  import javax.swing.JLabel;
30  import javax.swing.JPanel;
31  import javax.swing.JOptionPane;
32  import javax.swing.JTextField;
33
34  import org.tigam.railcab.algoritme.*;
35
36
37  //Het eigenlijke scherm!
38  /**
39   * @author dareal
40   *
41   */
42  class Paneel extends JPanel implements Observer{
43
44      protected boolean simulatieGestart = false, simulatieStatus;
45      private JButton start, stop, pauzeer, versnel, invoegen, moment;
46      protected JButton t1_kn, t2_kn, t3_kn, t4_kn, t5_kn, t6_kn, t7_kn, t8_kn,
47      s1_kn, s2_kn, s3_kn, s4_kn, s5_kn, s6_kn, s7_kn, s8_kn;
48      public JLabel aant_reiziger, vert_punt, bestemming, status;
49      protected JLabel hBaan, s1_label, s2_label, s3_label, s4_label, s5_label, s6_label, s7_label, s8_label, s_NaamLabel1,
50      s_NaamLabel2, s_NaamLabel3, s_NaamLabel4 , s_NaamLabel5, s_NaamLabel6, s_NaamLabel7, s_NaamLabel8;;
51      private JTextField aantal, tijdVeld;
52      private JComboBox vertrek, aankomst;
53      public ImageIcon achtergrond, hoofdbaan, taxi_img, station_img, station_img_vert,
54      s123_img, s4_img, s567_img, s8_img;
55      private Taxi taxi1, taxi2, taxi3, taxi4, taxi5, taxi6, taxi7, taxi8;
56      private int i,y,z = 0, sec , min, uur, aantalStation;
57      private static final int LIMIET_REIZIGER = 101;
58      private Controller controller;
59      public String pos;
60      private int tijd;
61      private int pexels, teller, n = 0;
62      public int r_inT, wT, rT, gB, g_wT, g_rT, taxi, tem;
63      private boolean simIsGestart = false, simIsPauze = false;
64
65
66      private JFrame frame;
67      private javax.swing.Timer timer;
68      private ArrayList<String> _station;
69      private ArrayListComboBoxModel station1, station2;
```

```
70      private BaanManager baanManager;
71      private ReisManager reisManager;
72      private int pixels = 0;
73      public Paneel(){
74
75
76          //Maak een arrayList aan voor de combobox
77          _station = new ArrayList<String>();
78          station1 = new ArrayListComboBoxModel(_station);
79          station2 = new ArrayListComboBoxModel(_station);
80          baanManager = Simulatie.getInstantie().getBaanManager();
81
82          //nieuw object van Controllerklasse
83          controller = new Controller(this);
84          reisManager = Simulatie.getInstantie().getReisManager();
85          frame = new JFrame();
86          setLayout( null );
87
88          timer = new javax.swing.Timer(1000, new ClockListener());//maakt timer aan
89
90          //haal plaatje op
91          achtergrond = new ImageIcon( "src/achtergrond.gif" );
92          taxi_img = new ImageIcon("src/taxi_stipje.gif");
93          station_img = new ImageIcon("src/station_knop.gif");
94          station_img_vert = new ImageIcon("src/station_knop_vert.gif");
95
96          s123_img = new ImageIcon("src/stations/station123.gif");
97          s4_img = new ImageIcon("src/stations/station4.gif");
98          s567_img = new ImageIcon("src/stations/station567.gif");
99          s8_img = new ImageIcon("src/stations/station8.gif");
100
101
102         //voeg de hoofdbaan toe
103         hoofdbaan = new ImageIcon( "src/hoofdbaan.gif" );
104         hBaan = new JLabel(hoofdbaan);
105         hBaan.setLocation(60,130);
106         hBaan.setSize(672, 333);
107         hBaan.setVisible( false );
108
109         //plak alle stations aan de baan vast
110         s1_label = new JLabel(s123_img);
111         s1_label.setLocation(110,97);
112         s1_label.setSize(80, 40);
113         s1_label.setVisible( false );
114
115         s2_label = new JLabel(s123_img);
116         s2_label.setLocation(351, 97);
117         s2_label.setSize(80, 40);
118         s2_label.setVisible( false );
119
120         s3_label = new JLabel(s123_img);
121         s3_label.setLocation(593, 97);
122         s3_label.setSize(80, 40);
123         s3_label.setVisible( false );
124
125         s4_label = new JLabel(s4_img);
126         s4_label.setLocation(714, 250);
127         s4_label.setSize(40, 80);
128         s4_label.setVisible( false );
129
130         s5_label = new JLabel(s567_img);
131         s5_label.setLocation(595, 454);
132         s5_label.setSize(80, 40);
133         s5_label.setVisible( false );
134
135         s6_label = new JLabel(s567_img);
136         s6_label.setLocation(357, 454);
137         s6_label.setSize(80, 40);
138         s6_label.setVisible( false );
139
140         s7_label = new JLabel(s567_img);
```

```
141         s7_label.setLocation(115, 454);
142         s7_label.setSize(80, 40);
143         s7_label.setVisible( false );
144 
145         s8_label = new JLabel(s8_img);
146         s8_label.setLocation(37, 255);
147         s8_label.setSize(40, 80);
148         s8_label.setVisible( false );
149 
150         //maak alle station naam label
151         s_NaamLabel1 = new JLabel();
152         s_NaamLabel1.setLocation(130, 135);
153         s_NaamLabel1.setSize(126, 25);
154         s_NaamLabel1.setVisible(false);
155 
156         s_NaamLabel2 = new JLabel();
157         s_NaamLabel2.setLocation(371, 135);
158         s_NaamLabel2.setSize(126, 25);
159         s_NaamLabel2.setVisible(false);
160 
161         s_NaamLabel3 = new JLabel();
162         s_NaamLabel3.setLocation(610, 135);
163         s_NaamLabel3.setSize(126, 25);
164         s_NaamLabel3.setVisible(false);
165 
166         s_NaamLabel4 = new JLabel();
167         s_NaamLabel4.setLocation(620, 255);
168         s_NaamLabel4.setSize(126, 25);
169         s_NaamLabel4.setVisible(false);
170 
171         s_NaamLabel5 = new JLabel();
172         s_NaamLabel5.setLocation(610, 430);
173         s_NaamLabel5.setSize(126, 25);
174         s_NaamLabel5.setVisible(false);
175 
176         s_NaamLabel6 = new JLabel();
177         s_NaamLabel6.setLocation(371, 430);
178         s_NaamLabel6.setSize(126, 25);
179         s_NaamLabel6.setVisible(false);
180 
181         s_NaamLabel7 = new JLabel();
182         s_NaamLabel7.setLocation(130, 430);
183         s_NaamLabel7.setSize(126, 25);
184         s_NaamLabel7.setVisible(false);
185 
186         s_NaamLabel8 = new JLabel();
187         s_NaamLabel8.setLocation(90, 255);
188         s_NaamLabel8.setSize(126, 25);
189         s_NaamLabel8.setVisible(false);
190 
191         //voeg de taxisknoppen toe
192         t1_kn = new JButton(taxi_img);
193         t1_kn.setSize(10,10);
194         t1_kn.addActionListener( new Taxi_1());
195         t1_kn.setVisible( false );
196 
197 
198         t2_kn = new JButton(taxi_img);
199         t2_kn.setSize(10,10);
200         t2_kn.addActionListener( new Taxi_2());
201         t2_kn.setVisible( false );
202 
203         t3_kn = new JButton(taxi_img);
204         t3_kn.setSize(10,10);
205         t3_kn.addActionListener( new Taxi_3());
206         t3_kn.setVisible( false );
207 
208         t4_kn = new JButton(taxi_img);
209         t4_kn.setSize(10,10);
210         t4_kn.addActionListener( new Taxi_4());
211         t4_kn.setVisible( false );
```

```
212
213         t5_kn = new JButton(taxi_img);
214         t5_kn.setSize(10,10);
215         t5_kn.addActionListener( new Taxi_5());
216         t5_kn.setVisible( false );
217
218         t6_kn = new JButton(taxi_img);
219         t6_kn.setSize(10,10);
220         t6_kn.addActionListener( new Taxi_6());
221         t6_kn.setVisible( false );
222
223         t7_kn = new JButton(taxi_img);
224         t7_kn.setSize(10,10);
225         t7_kn.addActionListener( new Taxi_7());
226         t7_kn.setVisible( false );
227
228         t8_kn = new JButton(taxi_img);
229         t8_kn.setSize(10,10);
230         t8_kn.addActionListener( new Taxi_8());
231         t8_kn.setVisible( false );
232
233         //voeg stationsknoppen toe
234         s1_kn = new JButton(station_img);
235         s1_kn.setSize(70,10);
236         s1_kn.setLocation(110, 77);
237         s1_kn.addActionListener( new Amstel());
238         s1_kn.setVisible( false );
239
240         s2_kn = new JButton(station_img);
241         s2_kn.setSize(70,10);
242         s2_kn.setLocation(357, 77);
243         s2_kn.addActionListener( new Sloterdijk());
244         s2_kn.setVisible( false );
245
246         s3_kn = new JButton(station_img);
247         s3_kn.setSize(70,10);
248         s3_kn.setLocation(593, 77);
249         s3_kn.addActionListener( new AmsterdamCS());
250         s3_kn.setVisible( false );
251
252         s4_kn = new JButton(station_img_vert);
253         s4_kn.setSize(10,70);
254         s4_kn.setLocation(765, 255);
255         s4_kn.addActionListener( new Lelylaan());
256         s4_kn.setVisible( false );
257
258         s5_kn = new JButton(station_img);
259         s5_kn.setSize(70,10);
260         s5_kn.setLocation(613, 510);
261         s5_kn.addActionListener( new Muiderpoort());
262         s5_kn.setVisible( false );
263
264         s6_kn = new JButton(station_img);
265         s6_kn.setSize(70,10);
266         s6_kn.setLocation(367, 510);
267         s6_kn.addActionListener( new ZuidWTC());
268         s6_kn.setVisible( false );
269
270         s7_kn = new JButton(station_img);
271         s7_kn.setSize(70,10);
272         s7_kn.setLocation(120, 510);
273         s7_kn.addActionListener( new Rai());
274         s7_kn.setVisible( false );
275
276         s8_kn = new JButton(station_img_vert);
277         s8_kn.setSize(10,70);
278         s8_kn.setLocation(15, 255);
279         s8_kn.addActionListener( new Duivendrecht());
280         s8_kn.setVisible( false );
281
282         //maak knoppen aan
```

```
283         start = new JButton("Simulatie Starten");
284         start.setLocation(136,650);
285         start.setSize(160,40);
286         start.addActionListener( new StartLabelWissel());
287 
288         pauzeer = new JButton ("Hervatten/Pauzeren");
289         pauzeer.setLocation(432,650);
290         pauzeer.setSize(160,40);
291         pauzeer.addActionListener( new HervatLabelWissel());
292 
293         versnel = new JButton ("Normaal/Versnellen");
294         versnel.setLocation(728,650);
295         versnel.setSize(160,40);
296         versnel.addActionListener( new VernselLabelWissel());
297 
298         invoegen = new JButton ("Invoegen");
299         invoegen.setLocation(750,610);
300         invoegen.setSize(120,25);
301         invoegen.addActionListener(new ReizigerInvoegen());
302 
303         moment = new JButton("MomentOpnameLaden");
304         moment.setLocation(830,420);
305         moment.setSize(163,25);
306         moment.addActionListener(new MomentOpenen());
307 
308         //maak textvakken aan
309         aantal = new JTextField();
310         aantal.setLocation(145,610);
311         aantal.setSize(120,25);
312         aantal.addActionListener(new GetAantalReis());
313 
314         //maak combobox aan
315         vertrek = new JComboBox(station1);
316         vertrek.setLocation(370,610);
317         vertrek.setSize(120,25);
318         vertrek.setSelectedIndex(-1);
319 
320         aankomst = new JComboBox(station2);
321         aankomst.setLocation(560,610);
322         aankomst.setSize(120,25);
323         aankomst.setSelectedIndex(-1);
324         voegToeStation();
325 
326         //maak timer aan
327         tijdVeld = new JTextField(5);
328         tijdVeld.setSize(120, 40);
329         tijdVeld.setLocation(400, 10);
330         tijdVeld.setEditable(false);
331         tijdVeld.setFont(new Font("sansserif", Font.PLAIN, 30));
332 
333         status = new JLabel("Gestopt");
334         status.setSize(100,25);
335         status.setLocation(790,43);
336 
337         //voeg stations_labels (de plaatjes) toe
338         add( status );
339         add (moment);
340 
341         add( s1_kn );
342         add( s2_kn );
343         add( s3_kn );
344         add( s4_kn );
345         add( s5_kn );
346         add( s6_kn );
347         add( s7_kn );
348         add( s8_kn );
349 
350         add( t1_kn );
351         add( t2_kn );
352         add( t3_kn );
353         add( t4_kn );
```

```
354             add( t5_kn );
355             add( t6_kn );
356             add( t7_kn );
357             add( t8_kn );
358 
359             add( s1_label );
360             add( s2_label );
361             add( s3_label );
362             add( s4_label );
363             add( s5_label );
364             add( s6_label );
365             add( s7_label );
366             add( s8_label );
367 
368             add(s_NaamLabel1);
369             add(s_NaamLabel2);
370             add(s_NaamLabel3);
371             add(s_NaamLabel4);
372             add(s_NaamLabel5);
373             add(s_NaamLabel6);
374             add(s_NaamLabel7);
375             add(s_NaamLabel8);
376 
377             add( hBaan );
378 
379             add( aantal );
380             add( start );
381             add( versnel );
382             add( pauzeer );
383             add( invoegen );
384             add( vertrek );
385             add( aankomst );
386             //add( tijdVeld );
387 
388 
389 
390         }
391 
392         /**
393          * Deze methode voeg de stations toe als de grootte van de array 0 is.
394          */
395         public void voegToeStation(){
396             if(_station.size()== 0){
397                 _station.add("Amstel");
398                 _station.add("Lelylaan");
399                 _station.add("Sloterdijk");
400                 _station.add("AmsterdamCS");
401                 _station.add("Muiderpoort");
402                 _station.add("ZuidWTC");
403                 _station.add("RAI");
404                 _station.add("Duivendrecht");
405             }
406 
407         }
408         /**
409          * Deze methode voegd lege string stations toe
410          */
411         public void voegToeLegeStation(){
412             _station.add("");
413             _station.add("");
414             _station.add("");
415             _station.add("");
416             _station.add("");
417             _station.add("");
418             _station.add("");
419             _station.add("");
420 
421         }
422 
423         /**
424          * Deze methode geeft aan de controller door hoeveel de aantal stations die zijn toegevoegd
```

```
425         * en het voeg ook de station namen toe in de controller klasse
426         */
427        public void setStationController(){
428            controller.setStations(_station.size());
429            String nm;
430            //probeer hiermee alle stations in te voeren in de controller klasse
431            for(int v = 0; v < _station.size(); v++){
432                nm = _station.get(v);
433                controller.setStation(nm);
434            }
435        }
436        /**
437         * Deze methode geeft aan de controller door de aantal taxi die zijn toegevoegd
438         * @param aantTaxi
439         */
440        public void setTaxi(int aantTaxi){
441            taxi = aantTaxi;
442            controller.setTaxi(taxi);
443            System.out.println(taxi);
444        }
445
446        /**
447         * Deze methode verwijder alle taxi in de array
448         */
449        public void verwijder(){
450            while(!(_station.size()== 0)){//deze mothode heeft betrekking op de verwijdering van de stations in de combobox
451                _station.remove(0);
452            }
453        }
454
455        /**
456         * Deze methode maakt de status van de simulatie true of false
457         * @param simulatie
458         */
459        public void simulatieGestart(boolean simulatie){
460            simulatieGestart = simulatie;
461        }
462
463        //Dit roep het scherm aan om meer taxis toe te voegen
464        public void voegMeerTaxiToe(){
465            String TaxiToevoegen = JOptionPane.showInputDialog(null, "Aantal taxi's:","Meerdere Taxi's Toevoegen in het simulatie",JOptionPane.YES_NO_CANCEL_OPTION);
466            int aantalTaxis = Integer.parseInt(TaxiToevoegen);
467            //hier worden de taxi in de simulatie toegevoegd
468
469        }
470
471        /**
472         * Deze methode blijft stations verwijderen vanaf  de 0-de index tot hij aan de opgegeven in de param voldoed.
473         * @param verwijder
474         */
475        public void verwijder(int verwijder){//deze mothode heeft betrekking op de verwijdering van de stations in de combobox
476            switch(verwijder){
477            case 8:
478                verwijder();
479                break;
480            case 7:
481                while(!(_station.size()== 7)){
482                    _station.remove(0);
483                }
484                break;
485            case 6:
486                while(!(_station.size()== 6)){
487                    _station.remove(0);
488                }
489                break;
490            case 5:
491                while(!(_station.size()== 5)){
492                    _station.remove(0);
493                }
494                break;
495            case 4:
```

```
496             while(!(_station.size()== 4)){
497                 _station.remove(0);
498             }
499             break;
500         case 3:
501             while(!(_station.size()== 3)){
502                 _station.remove(0);
503             }
504             break;
505         case 2:
506             while(!(_station.size()== 2)){
507                 _station.remove(0);
508             }
509             break;
510         case 1:
511             while(!(_station.size()== 1)){
512                 _station.remove(0);
513             }
514             break;
515         }
516     }
517 
518     /**
519      * Deze methode represeteert de gemiddil statiestiek met behulp van een optionPane
520      */
521     public void gemiddelStatistiek(){
522         String bericht = controller.getGemStatistieken();
523         JOptionPane optionPane = new GemmidelstatistiekScherm();
524         optionPane.setMessage(bericht);
525         optionPane.setMessageType(JOptionPane.INFORMATION_MESSAGE);
526         JDialog dialog = optionPane.createDialog(null, "Gemiddelde statistieken");
527         dialog.setVisible(true);
528     }
529 
530     /**
531      * Deze code roep de methode moment opslaan in de controller klasse. om dat te doen
532      */
533     public void momentOpnameOpslaan(){
534         controller.momentOpnameOpslaan();
535 
536     }
537     //methods voor de stations
538 
539     /**
540      * Deze methode zorgt ervoor dat er 2 stations op de baan getekend worden
541      */
542     public void tweeStations(){
543         s1_label.setVisible( true );
544         s1_kn.setVisible( true );
545         s5_label.setVisible( true );
546         s5_kn.setVisible( true );
547         voegToeStation();
548         verwijder(2);
549         setStationController();
550         aantalStation = 2;// geeft de waarde aan voor de switch case van de statistiek
551         stationLabel(2);
552         //1, 5
553     }
554     /**
555      * Deze methode zorgt ervoor dat er 3 stations op de baan getekend worden
556      */
557     public void drieStations(){
558         s2_label.setVisible( true );
559         s2_kn.setVisible( true );
560         s5_label.setVisible( true );
561         s5_kn.setVisible( true );
562         s7_label.setVisible( true );
563         s7_kn.setVisible( true );
564         voegToeStation();
565         verwijder(3);
566         setStationController();
```

```
567             aantalStation = 3;
568             stationLabel(3);
569 
570 
571             //2, 5, 7
572         }
573         /**
574          * Deze methode zorgt ervoor dat er 4 stations op de baan getekend worden
575          */
576         public void vierStations(){
577             s1_label.setVisible( true );
578             s1_kn.setVisible( true );
579             s3_label.setVisible( true );
580             s3_kn.setVisible( true );
581             s5_label.setVisible( true );
582             s5_kn.setVisible( true );
583             s7_label.setVisible( true );
584             s7_kn.setVisible( true );
585             voegToeStation();
586             verwijder(4);
587             setStationController();
588             aantalStation = 4;
589             stationLabel(4);
590 
591             //1, 3, 5, 7
592         }
593         /**
594          * Deze methode zorgt ervoor dat er 5 stations op de baan getekend worden
595          */
596         public void vijfStations(){
597             s2_label.setVisible( true );
598             s2_kn.setVisible( true );
599             s4_label.setVisible( true );
600             s4_kn.setVisible( true );
601             s5_label.setVisible( true );
602             s5_kn.setVisible( true );
603             s7_label.setVisible( true );
604             s7_kn.setVisible( true );
605             s8_label.setVisible( true );
606             s8_kn.setVisible( true );
607             voegToeStation();
608             verwijder(5);
609             setStationController();
610             aantalStation = 5;
611             stationLabel(5);
612 
613             //2, 4, 5, 7, 8
614         }
615         /**
616          * Deze methode zorgt ervoor dat er 6 stations op de baan getekend worden
617          */
618         public void zesStations(){
619             s1_label.setVisible( true );
620             s1_kn.setVisible( true );
621             s3_label.setVisible( true );
622             s3_kn.setVisible( true );
623             s4_label.setVisible( true );
624             s4_kn.setVisible( true );
625             s5_label.setVisible( true );
626             s5_kn.setVisible( true );
627             s7_label.setVisible( true );
628             s7_kn.setVisible( true );
629             s8_label.setVisible( true );
630             s8_kn.setVisible( true );
631             voegToeStation();
632             verwijder(6);
633             setStationController();
634             aantalStation = 6;
635             stationLabel(6);
636 
637 
```

```
638         //1, 3, 4, 5, 7, 8
639     }
640     /**
641      * Deze methode zorgt ervoor dat er 7 stations op de baan getekend worden
642      */
643     public void zevenStations(){
644         s1_label.setVisible( true );
645         s1_kn.setVisible( true );
646         s2_label.setVisible( true );
647         s2_kn.setVisible( true );
648         s3_label.setVisible( true );
649         s3_kn.setVisible( true );
650         s4_label.setVisible( true );
651         s4_kn.setVisible( true );
652         s5_label.setVisible( true );
653         s5_kn.setVisible( true );
654         s7_label.setVisible( true );
655         s7_kn.setVisible( true );
656         s8_label.setVisible( true );
657         s8_kn.setVisible( true );
658         voegToeStation();
659         verwijder(7);
660         setStationController();
661         aantalStation = 7;
662         stationLabel(7);
663 
664 
665         //1, 2, 3, 4, 5, 7, 8
666     }
667     /**
668      * Deze methode zorgt ervoor dat er 8 stations op de baan getekend worden
669      */
670     public void achtStations(){
671         s1_label.setVisible( true );
672         s1_kn.setVisible( true );
673         s2_label.setVisible( true );
674         s2_kn.setVisible( true );
675         s3_label.setVisible( true );
676         s3_kn.setVisible( true );
677         s4_label.setVisible( true );
678         s4_kn.setVisible( true );
679         s5_label.setVisible( true );
680         s5_kn.setVisible( true );
681         s6_label.setVisible( true );
682         s6_kn.setVisible( true );
683         s7_label.setVisible( true );
684         s7_kn.setVisible( true );
685         s8_label.setVisible( true );
686         s8_kn.setVisible( true );
687         voegToeStation();
688         setStationController();
689         aantalStation = 8;
690         stationLabel(8);
691 
692         //alles
693     }
694     /**
695      *
696      * Deze methode zal aan de hand van de doorgekregen absolute en zijpositie berekenen wat
697      * de positie op het scherm is, per taxi.
698      * @param taxi
699      * @param meters
700      * @param zij
701      * @param opStat
702      * @param naarStat
703      * @param vanStat
704      * @param test
705      * @param versnel
706      */
707     public void setLocation(int taxi, int meters, int zij, boolean opStat, boolean naarStat, boolean vanStat, int test, int versnel){
708         pixels = meters; //meters naar pixels
```

```
709         if (taxi == 1){
710
711             //eerste stuk horizontaal
712             if (meters <= 640 && zij == 0){
713                 t1_kn.setLocation(70+pixels,130);
714             }
715
716             if( zij >0 && zij <= 40 && meters <=640){ //naar het station
717                 t1_kn.setLocation(meters+zij+70,130-zij);
718             }
719
720             if( zij > 40 && zij < 80 && meters <=640){ //van het station
721                 t1_kn.setLocation(meters+zij+70,90+(zij-40));
722             }
723
724             //eerste stuk vertikaal
725             if( meters > 640 && meters <= 960  && zij == 0){
726                 t1_kn.setLocation(710,130+pixels-640);
727             }
728
729             if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
730                 t1_kn.setLocation(710+zij,250+zij);
731             }
732
733             if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
734                 t1_kn.setLocation(790-(zij), 250+zij);
735             }
736
737             //tweed stuk horizontaal
738             if( meters > 960 && meters <= 1600  && zij == 0){
739                 t1_kn.setLocation(710-(pixels-960),450);
740             }
741             if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
742                 t1_kn.setLocation(710-((meters+zij)-960),450+zij);
743             }
744             if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
745                 t1_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
746             }
747
748
749             //tweede stuk vertikaal
750             if( meters >1600 && meters <=1920  && zij == 0){
751                 t1_kn.setLocation(70,450-(pixels-1600));
752             }
753             if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
754                 t1_kn.setLocation(70-zij,330-zij);
755             }
756             if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
757                 t1_kn.setLocation(70+(zij-80), 330-zij);
758             }
759
760         }
761         //taxi 2
762         if (taxi == 2){
763
764             //eerste stuk horizontaal
765             if (meters <= 640 && zij == 0){
766                 t2_kn.setLocation(70+pixels,130);
767             }
768
769             if( zij >0 && zij <= 40 && meters <=640){ //naar het station
770                 t2_kn.setLocation(meters+zij+70,130-zij);
771             }
772
773             if( zij > 40 && zij < 80 && meters <=640){ //van het station
774                 t2_kn.setLocation(meters+zij+70,90+(zij-40));
775             }
776
777             //eerste stuk vertikaal
778             if( meters > 640 && meters <= 960  && zij == 0){
779                 t2_kn.setLocation(710,130+pixels-640);
```

```
780             }
781
782             if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
783                 t2_kn.setLocation(710+zij,250+zij);
784             }
785
786             if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
787                 t2_kn.setLocation(790-(zij), 250+zij);
788             }
789
790             //tweed stuk horizontaal
791             if( meters > 960 && meters <= 1600  && zij == 0){
792                 t2_kn.setLocation(710-(pixels-960),450);
793             }
794             if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
795                 t2_kn.setLocation(710-((meters+zij)-960),450+zij);
796             }
797             if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
798                 t2_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
799             }
800
801
802             //tweede stuk vertikaal
803             if( meters >1600 && meters <=1920  && zij == 0){
804                 t2_kn.setLocation(70,450-(pixels-1600));
805             }
806             if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
807                 t2_kn.setLocation(70-zij,330-zij);
808             }
809             if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
810                 t2_kn.setLocation(70+(zij-80), 330-zij);
811             }
812
813         }
814
815         if (taxi == 3 ){
816
817             //eerste stuk horizontaal
818             if (meters <= 640 && zij == 0){
819                 t3_kn.setLocation(70+pixels,130);
820             }
821
822             if( zij >0 && zij <= 40 && meters <=640){ //naar het station
823                 t3_kn.setLocation(meters+zij+70,130-zij);
824             }
825
826             if( zij > 40 && zij < 80 && meters <=640){ //van het station
827                 t3_kn.setLocation(meters+zij+70,90+(zij-40));
828             }
829
830             //eerste stuk vertikaal
831             if( meters > 640 && meters <= 960  && zij == 0){
832                 t3_kn.setLocation(710,130+pixels-640);
833             }
834
835             if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
836                 t3_kn.setLocation(710+zij,250+zij);
837             }
838
839             if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
840                 t3_kn.setLocation(790-(zij), 250+zij);
841             }
842
843             //tweed stuk horizontaal
844             if( meters > 960 && meters <= 1600  && zij == 0){
845                 t3_kn.setLocation(710-(pixels-960),450);
846             }
847             if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
848                 t3_kn.setLocation(710-((meters+zij)-960),450+zij);
849             }
850             if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
```

```
851                 t3_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
852             }
853 
854 
855             //tweede stuk vertikaal
856             if( meters >1600 && meters <=1920  && zij == 0){
857                 t3_kn.setLocation(70,450-(pixels-1600));
858             }
859             if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
860                 t3_kn.setLocation(70-zij,330-zij);
861             }
862             if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
863                 t3_kn.setLocation(70+(zij-80), 330-zij);
864             }
865 
866         }
867 
868         if (taxi == 4){
869 
870             //eerste stuk horizontaal
871             if (meters <= 640 && zij == 0){
872                 t4_kn.setLocation(70+pixels,130);
873             }
874 
875             if( zij >0 && zij <= 40 && meters <=640){ //naar het station
876                 t4_kn.setLocation(meters+zij+70,130-zij);
877             }
878 
879             if( zij > 40 && zij < 80 && meters <=640){ //van het station
880                 t4_kn.setLocation(meters+zij+70,90+(zij-40));
881             }
882 
883             //eerste stuk vertikaal
884             if( meters > 640 && meters <= 960  && zij == 0){
885                 t4_kn.setLocation(710,130+pixels-640);
886             }
887 
888             if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
889                 t4_kn.setLocation(710+zij,250+zij);
890             }
891 
892             if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
893                 t4_kn.setLocation(790-(zij), 250+zij);
894             }
895 
896             //tweed stuk horizontaal
897             if( meters > 960 && meters <= 1600  && zij == 0){
898                 t4_kn.setLocation(710-(pixels-960),450);
899             }
900             if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
901                 t4_kn.setLocation(710-((meters+zij)-960),450+zij);
902             }
903             if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
904                 t4_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
905             }
906 
907 
908             //tweede stuk vertikaal
909             if( meters >1600 && meters <=1920  && zij == 0){
910                 t4_kn.setLocation(70,450-(pixels-1600));
911             }
912             if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
913                 t4_kn.setLocation(70-zij,330-zij);
914             }
915             if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
916                 t4_kn.setLocation(70+(zij-80), 330-zij);
917             }
918 
919         }
920 
921         if (taxi == 5){
```

```
922
923             //eerste stuk horizontaal
924             if (meters <= 640 && zij == 0){
925                 t5_kn.setLocation(70+pixels,130);
926             }
927
928             if( zij >0 && zij <= 40 && meters <=640){ //naar het station
929                 t5_kn.setLocation(meters+zij+70,130-zij);
930             }
931
932             if( zij > 40 && zij < 80 && meters <=640){ //van het station
933                 t5_kn.setLocation(meters+zij+70,90+(zij-40));
934             }
935
936             //eerste stuk vertikaal
937             if( meters > 640 && meters <= 960  && zij == 0){
938                 t5_kn.setLocation(710,130+pixels-640);
939             }
940
941             if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
942                 t5_kn.setLocation(710+zij,250+zij);
943             }
944
945             if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
946                 t5_kn.setLocation(790-(zij), 250+zij);
947             }
948
949             //tweed stuk horizontaal
950             if( meters > 960 && meters <= 1600  && zij == 0){
951                 t5_kn.setLocation(710-(pixels-960),450);
952             }
953             if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
954                 t5_kn.setLocation(710-((meters+zij)-960),450+zij);
955             }
956             if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
957                 t5_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
958             }
959
960
961             //tweede stuk vertikaal
962             if( meters >1600 && meters <=1920  && zij == 0){
963                 t5_kn.setLocation(70,450-(pixels-1600));
964             }
965             if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
966                 t5_kn.setLocation(70-zij,330-zij);
967             }
968             if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
969                 t5_kn.setLocation(70+(zij-80), 330-zij);
970             }
971
972         }
973         if (taxi == 6){
974
975             //eerste stuk horizontaal
976             if (meters <= 640 && zij == 0){
977                 t6_kn.setLocation(70+pixels,130);
978             }
979
980             if( zij >0 && zij <= 40 && meters <=640){ //naar het station
981                 t6_kn.setLocation(meters+zij+70,130-zij);
982             }
983
984             if( zij > 40 && zij < 80 && meters <=640){ //van het station
985                 t6_kn.setLocation(meters+zij+70,90+(zij-40));
986             }
987
988             //eerste stuk vertikaal
989             if( meters > 640 && meters <= 960  && zij == 0){
990                 t6_kn.setLocation(710,130+pixels-640);
991             }
992
```

```
993             if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
994                 t6_kn.setLocation(710+zij,250+zij);
995             }
996 
997             if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
998                 t6_kn.setLocation(790-(zij), 250+zij);
999             }
1000 
1001            //tweed stuk horizontaal
1002            if( meters > 960 && meters <= 1600  && zij == 0){
1003                t6_kn.setLocation(710-(pixels-960),450);
1004            }
1005            if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
1006                t6_kn.setLocation(710-((meters+zij)-960),450+zij);
1007            }
1008            if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
1009                t6_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
1010            }
1011 
1012 
1013            //tweede stuk vertikaal
1014            if( meters >1600 && meters <=1920  && zij == 0){
1015                t6_kn.setLocation(70,450-(pixels-1600));
1016            }
1017            if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
1018                t6_kn.setLocation(70-zij,330-zij);
1019            }
1020            if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
1021                t6_kn.setLocation(70+(zij-80), 330-zij);
1022            }
1023 
1024        }
1025 
1026        if (taxi == 7){
1027 
1028            //eerste stuk horizontaal
1029            if (meters <= 640 && zij == 0){
1030                t7_kn.setLocation(70+pixels,130);
1031            }
1032 
1033            if( zij >0 && zij <= 40 && meters <=640){ //naar het station
1034                t7_kn.setLocation(meters+zij+70,130-zij);
1035            }
1036 
1037            if( zij > 40 && zij < 80 && meters <=640){ //van het station
1038                t7_kn.setLocation(meters+zij+70,90+(zij-40));
1039            }
1040 
1041            //eerste stuk vertikaal
1042            if( meters > 640 && meters <= 960  && zij == 0){
1043                t7_kn.setLocation(710,130+pixels-640);
1044            }
1045 
1046            if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
1047                t7_kn.setLocation(710+zij,250+zij);
1048            }
1049 
1050            if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
1051                t7_kn.setLocation(790-(zij), 250+zij);
1052            }
1053 
1054            //tweed stuk horizontaal
1055            if( meters > 960 && meters <= 1600  && zij == 0){
1056                t7_kn.setLocation(710-(pixels-960),450);
1057            }
1058            if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
1059                t7_kn.setLocation(710-((meters+zij)-960),450+zij);
1060            }
1061            if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
1062                t7_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
1063            }
```

```
1064
1065
1066            //tweede stuk vertikaal
1067            if( meters >1600 && meters <=1920  && zij == 0){
1068                t7_kn.setLocation(70,450-(pixels-1600));
1069            }
1070            if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
1071                t7_kn.setLocation(70-zij,330-zij);
1072            }
1073            if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
1074                t7_kn.setLocation(70+(zij-80), 330-zij);
1075            }
1076
1077        }
1078
1079        if (taxi == 8){
1080
1081            //eerste stuk horizontaal
1082            if (meters <= 640 && zij == 0){
1083                t8_kn.setLocation(70+pixels,130);
1084            }
1085
1086            if( zij >0 && zij <= 40 && meters <=640){ //naar het station
1087                t8_kn.setLocation(meters+zij+70,130-zij);
1088            }
1089
1090            if( zij > 40 && zij < 80 && meters <=640){ //van het station
1091                t8_kn.setLocation(meters+zij+70,90+(zij-40));
1092            }
1093
1094            //eerste stuk vertikaal
1095            if( meters > 640 && meters <= 960  && zij == 0){
1096                t8_kn.setLocation(710,130+pixels-640);
1097            }
1098
1099            if( zij >0 && zij <= 40 && meters > 640 && meters <= 960){ //naar het station
1100                t8_kn.setLocation(710+zij,250+zij);
1101            }
1102
1103            if( zij > 40 && zij < 80 && meters > 640 && meters <= 960){ //van het station
1104                t8_kn.setLocation(790-(zij), 250+zij);
1105            }
1106
1107            //tweed stuk horizontaal
1108            if( meters > 960 && meters <= 1600  && zij == 0){
1109                t8_kn.setLocation(710-(pixels-960),450);
1110            }
1111            if( zij >0 && zij <= 40 && meters > 960 && meters <= 1600){ //naar het station
1112                t8_kn.setLocation(710-((meters+zij)-960),450+zij);
1113            }
1114            if( zij > 40 && zij < 80 && meters > 960 && meters <= 1600){ //van het station
1115                t8_kn.setLocation(710-(meters-960)-(zij), 450-(zij-80));
1116            }
1117
1118
1119            //tweede stuk vertikaal
1120            if( meters >1600 && meters <=1920  && zij == 0){
1121                t8_kn.setLocation(70,450-(pixels-1600));
1122            }
1123            if( zij >0 && zij <= 40 && meters > 1600 && meters <= 1920){ //naar het station
1124                t8_kn.setLocation(70-zij,330-zij);
1125            }
1126            if( zij > 40 && zij < 80 && meters > 1600 && meters <= 1920){ //van het station
1127                t8_kn.setLocation(70+(zij-80), 330-zij);
1128            }
1129
1130        }
1131    }
1132
1133
1134    /**
```

```
1135        *
1136        * Hij zet alle taxi's op hun initiele plek, waarna ze vervolgens verplaatsen kunnen.
1137        * hij berekend de initiele plek aan de hand van de absolute positie.
1138        * @param absolute
1139        * @param zij
1140        * @param ID
1141        */
1142       public void setInitLoc(double absolute, double zij, int ID){
1143           int ab = 0;
1144           int z = 0;
1145
1146           ab = new Double((absolute)/12.5).intValue();
1147           z = new Double((zij)/12.5).intValue();
1148
1149
1150           System.out.println("Taxi: " + ID);
1151           System.out.println(ab);
1152           System.out.println(z);
1153
1154           if(ID == 1){
1155               if(ab <= 640){
1156                   t1_kn.setLocation(110+ab,50+z);
1157               }
1158               if(ab> 640 && ab <= 960 ){
1159                   t1_kn.setLocation(750,130+(ab-640)+40);
1160               }
1161               if(ab> 960 && ab <=1600){
1162                   t1_kn.setLocation(710-(ab-920),450+z);
1163               }
1164               if(ab>1600 && ab <1920){
1165                   t1_kn.setLocation(30,450-(ab-1560));
1166               }
1167           }
1168           if(ID == 2){
1169
1170               if(ab <= 640){
1171                   t2_kn.setLocation(110+ab,50+z);
1172               }
1173               if(ab> 640 && ab <= 960 ){
1174                   t2_kn.setLocation(750,130+(ab-640)+40);
1175               }
1176               if(ab> 960 && ab <=1600){
1177                   t2_kn.setLocation(710-(ab-920),450+z);
1178               }
1179               if(ab>1600 && ab <1920){
1180                   t2_kn.setLocation(30,450-(ab-1560));
1181               }
1182           }
1183
1184           if(ID == 3){
1185               if(ab <= 640){
1186                   t3_kn.setLocation(110+ab,50+z);
1187               }
1188               if(ab> 640 && ab <= 960 ){
1189                   t3_kn.setLocation(750,130+(ab-640)+40);
1190               }
1191               if(ab> 960 && ab <=1600){
1192                   t3_kn.setLocation(710-(ab-920),450+z);
1193               }
1194               if(ab>1600 && ab <1920){
1195                   t3_kn.setLocation(30,450-(ab-1560));
1196               }
1197           }
1198           if(ID == 4){
1199               if(ab <= 640){
1200                   t4_kn.setLocation(110+ab,50+z);
1201               }
1202               if(ab> 640 && ab <= 960 ){
1203                   t4_kn.setLocation(750,130+(ab-640)+40);
1204               }
1205               if(ab> 960 && ab <=1600){
```

```
1206                t4_kn.setLocation(710-(ab-920),450+z);
1207            }
1208            if(ab>1600 && ab <1920){
1209                t4_kn.setLocation(30,450-(ab-1560));
1210            }
1211        }
1212        if(ID == 5){
1213            if(ab <= 640){
1214                t5_kn.setLocation(110+ab,50+z);
1215            }
1216            if(ab> 640 && ab <= 960 ){
1217                t5_kn.setLocation(750,130+(ab-640)+40);
1218            }
1219            if(ab> 960 && ab <=1600){
1220                t5_kn.setLocation(710-(ab-920),450+z);
1221            }
1222            if(ab>1600 && ab <1920){
1223                t5_kn.setLocation(30,450-(ab-1560));
1224            }
1225        }
1226        if(ID == 6){
1227            if(ab <= 640){
1228                t6_kn.setLocation(110+ab,50+z);
1229            }
1230            if(ab> 640 && ab <= 960 ){
1231                t6_kn.setLocation(750,130+(ab-640)+40);
1232            }
1233            if(ab> 960 && ab <=1600){
1234                t6_kn.setLocation(710-(ab-920),450+z);
1235            }
1236            if(ab>1600 && ab <1920){
1237                t6_kn.setLocation(30,450-(ab-1560));
1238            }
1239        }
1240        if(ID == 7){
1241            if(ab <= 640){
1242                t7_kn.setLocation(110+ab,50+z);
1243            }
1244            if(ab> 640 && ab <= 960 ){
1245                t7_kn.setLocation(750,130+(ab-640)+40);
1246            }
1247            if(ab> 960 && ab <=1600){
1248                t7_kn.setLocation(710-(ab-920),450+z);
1249            }
1250            if(ab>1600 && ab <1920){
1251                t7_kn.setLocation(30,450-(ab-1560));
1252            }
1253        }
1254        if(ID == 8){
1255            if(ab <= 640){
1256                t8_kn.setLocation(110+ab,50+z);
1257            }
1258            if(ab> 640 && ab <= 960 ){
1259                t8_kn.setLocation(750,130+(ab-640)+40);
1260            }
1261            if(ab> 960 && ab <=1600){
1262                t8_kn.setLocation(710-(ab-920),450+z);
1263            }
1264            if(ab>1600 && ab <1920){
1265                t8_kn.setLocation(30,450-(ab-1560));
1266            }
1267        }
1268    }
1269    //methodes voor de taxi's
1270
1271    /**
1272     * Tekend 1 taxi
1273     */
1274    public void eenTaxi(){
1275        t1_kn.setVisible( true );
1276    }
```

```
1277      /**
1278       * Tekend 2 taxi's
1279       */
1280      public void tweeTaxi(){
1281          t1_kn.setVisible( true );
1282          t2_kn.setVisible( true );
1283
1284      }
1285      /**
1286       * Tekend 3 taxi's
1287       */
1288      public void drieTaxi(){
1289          t1_kn.setVisible( true );
1290          t2_kn.setVisible( true );
1291          t3_kn.setVisible( true );
1292
1293      }
1294      /**
1295       * Tekend 4 taxi's
1296       */
1297      public void vierTaxi(){
1298          t1_kn.setVisible( true );
1299          t2_kn.setVisible( true );
1300          t3_kn.setVisible( true );
1301          t4_kn.setVisible( true );
1302
1303      }
1304      /**
1305       * Tekend 5 taxi's
1306       */
1307      public void vijfTaxi(){
1308          t1_kn.setVisible( true );
1309          t2_kn.setVisible( true );
1310          t3_kn.setVisible( true );
1311          t4_kn.setVisible( true );
1312          t5_kn.setVisible( true );
1313
1314      }
1315      /**
1316       * Tekend 6 taxi's
1317       */
1318      public void zesTaxi(){
1319          t1_kn.setVisible( true );
1320          t2_kn.setVisible( true );
1321          t3_kn.setVisible( true );
1322          t4_kn.setVisible( true );
1323          t5_kn.setVisible( true );
1324          t6_kn.setVisible( true );
1325
1326      }
1327      /**
1328       * Tekend 7 taxi's
1329       */
1330      public void zevenTaxi(){
1331          t1_kn.setVisible( true );
1332          t2_kn.setVisible( true );
1333          t3_kn.setVisible( true );
1334          t4_kn.setVisible( true );
1335          t5_kn.setVisible( true );
1336          t6_kn.setVisible( true );
1337          t7_kn.setVisible( true );
1338
1339      }
1340      /**
1341       * Tekend 8 taxi's
1342       */
1343      public void achtTaxi(){
1344          t1_kn.setVisible( true );
1345          t2_kn.setVisible( true );
1346          t3_kn.setVisible( true );
1347          t4_kn.setVisible( true );
```

```
1348          t5_kn.setVisible( true );
1349          t6_kn.setVisible( true );
1350          t7_kn.setVisible( true );
1351          t8_kn.setVisible( true );
1352
1353      }
1354      /**
1355       * Opend een nieuw object van TekenSpoor, met this als argument, dat wil zeggen, de constructor van paneel.
1356       */
1357      public void openPaneel(){
1358          new TekenSpoor(this);
1359
1360      }
1361      /**
1362       * Deze methode vult de label text in met de behoordestation naam die afkomstig is vanuit de controller. Om dat de doen maak
1363       * hij gebruik van de aantalStation parameter die hij krijgt als je de aantal statioon invoerd
1364       * @param aantalStation
1365       */
1366      public void stationLabel(int aantalStation){
1367          switch(aantalStation){
1368          case 8:
1369              s_NaamLabel1.setText(controller.getNaam(1));
1370              s_NaamLabel2.setText(controller.getNaam(2));
1371              s_NaamLabel3.setText(controller.getNaam(3));
1372              s_NaamLabel4.setText(controller.getNaam(4));
1373              s_NaamLabel5.setText(controller.getNaam(5));
1374              s_NaamLabel6.setText(controller.getNaam(6));
1375              s_NaamLabel7.setText(controller.getNaam(7));
1376              s_NaamLabel8.setText(controller.getNaam(8));
1377              break;
1378          case 7:
1379              s_NaamLabel1.setText(controller.getNaam(1));
1380              s_NaamLabel2.setText(controller.getNaam(2));
1381              s_NaamLabel3.setText(controller.getNaam(3));
1382              s_NaamLabel4.setText(controller.getNaam(4));
1383              s_NaamLabel5.setText(controller.getNaam(5));
1384              s_NaamLabel7.setText(controller.getNaam(6));
1385              s_NaamLabel8.setText(controller.getNaam(7));
1386              break;
1387          case 6:
1388              s_NaamLabel1.setText(controller.getNaam(1));
1389              s_NaamLabel3.setText(controller.getNaam(2));
1390              s_NaamLabel4.setText(controller.getNaam(3));
1391              s_NaamLabel5.setText(controller.getNaam(4));
1392              s_NaamLabel7.setText(controller.getNaam(5));
1393              s_NaamLabel8.setText(controller.getNaam(6));
1394              break;
1395          case 5:
1396              s_NaamLabel2.setText(controller.getNaam(1));
1397              s_NaamLabel4.setText(controller.getNaam(2));
1398              s_NaamLabel5.setText(controller.getNaam(3));
1399              s_NaamLabel7.setText(controller.getNaam(4));
1400              s_NaamLabel8.setText(controller.getNaam(5));
1401              break;
1402          case 4:
1403              s_NaamLabel1.setText(controller.getNaam(1));
1404              s_NaamLabel3.setText(controller.getNaam(2));
1405              s_NaamLabel5.setText(controller.getNaam(3));
1406              s_NaamLabel7.setText(controller.getNaam(4));
1407              break;
1408          case 3:
1409              s_NaamLabel2.setText(controller.getNaam(1));
1410              s_NaamLabel5.setText(controller.getNaam(2));
1411              s_NaamLabel7.setText(controller.getNaam(3));
1412              break;
1413          case 2:
1414              s_NaamLabel1.setText(controller.getNaam(1));
1415              s_NaamLabel5.setText(controller.getNaam(2));
1416              break;
1417          }
1418      }
```

```
1419      /**
1420       * Deze methode zorg ervopor dat je de label kan zien op het paneel
1421       */
1422      public void zetLabelOpTrue(){
1423          s_NaamLabel1.setVisible(true);
1424          s_NaamLabel2.setVisible(true);
1425          s_NaamLabel3.setVisible(true);
1426          s_NaamLabel4.setVisible(true);
1427          s_NaamLabel5.setVisible(true);
1428          s_NaamLabel6.setVisible(true);
1429          s_NaamLabel7.setVisible(true);
1430          s_NaamLabel8.setVisible(true);
1431      }
1432      /**
1433       * Deze methode zet de label op spatie
1434       */
1435      public void zetLabelTextOpNull(){
1436          s_NaamLabel1.setText(" ");
1437          s_NaamLabel2.setText(" ");
1438          s_NaamLabel3.setText(" ");
1439          s_NaamLabel4.setText(" ");
1440          s_NaamLabel5.setText(" ");
1441          s_NaamLabel6.setText(" ");
1442          s_NaamLabel7.setText(" ");
1443          s_NaamLabel8.setText(" ");
1444      }
1445      /**
1446       * Deze methode zorg ervoor dat je de label onzichtbaar worden
1447       */
1448      public void zelLabelOpFalse(){
1449          s_NaamLabel1.setVisible(false);
1450          s_NaamLabel2.setVisible(false);
1451          s_NaamLabel3.setVisible(false);
1452          s_NaamLabel4.setVisible(false);
1453          s_NaamLabel5.setVisible(false);
1454          s_NaamLabel6.setVisible(false);
1455          s_NaamLabel7.setVisible(false);
1456          s_NaamLabel8.setVisible(false);
1457      }
1458      public void startTimer(){//start timer
1459          timer.start();
1460      }
1461      public void stopTimer(){//stop timer
1462          timer.stop();
1463          uur = 0;
1464          min = 0;
1465          sec = 0;
1466
1467      }
1468      public void setLocation(){
1469
1470      }
1471      public void pauzeTimer(){
1472          timer.stop();
1473      }
1474      public void VersnelOfNormaliceerTimer(int i){
1475          if(i != 1000){
1476              timer.setDelay(controller.versnelTimer());
1477          }
1478          else {
1479              timer.setDelay(1000);
1480          }
1481
1482      }
1483      public String TimertoString (){
1484
1485          return tijdVeld.getText();
1486      }
1487      /**
1488       * Deze methode voegt de stations toe. Wordt voor momentladen gebruikt
1489       * @param station
```

```
1490      */
1491     public void voegStationsToe(int station){
1492 
1493         switch (station) {
1494         case 2: tweeStations(); break;
1495         case 3: drieStations(); break;
1496         case 4: vierStations(); break;
1497         case 5: vijfStations(); break;
1498         case 6: zesStations(); break;
1499         case 7: zevenStations(); break;
1500         case 8: achtStations(); break;
1501         default: break;
1502         }
1503     }
1504     /**
1505      * Deze methode voeg de aantal taxi toe... Wordt vooor moment opname gebruikt
1506      * @param t_axi
1507      */
1508     public void voegTaxiToe(int t_axi){
1509         switch (t_axi) {
1510         case 1: eenTaxi();
1511         setTaxi(t_axi);
1512         break;
1513         case 2: tweeTaxi();
1514         setTaxi(t_axi);
1515         break;
1516         case 3: drieTaxi();
1517         setTaxi(t_axi);
1518         break;
1519         case 4: vierTaxi();
1520         setTaxi(t_axi);
1521         break;
1522         case 5: vijfTaxi();
1523         setTaxi(t_axi);
1524         break;
1525         case 6: zesTaxi();
1526         setTaxi(t_axi);
1527         break;
1528         case 7: zevenTaxi();
1529         setTaxi(t_axi);
1530         break;
1531         case 8: achtTaxi();
1532         setTaxi(t_axi);
1533         break;
1534         default: break;
1535         }
1536     }
1537 
1538     /**
1539      * Deze methode wordt gebruikt om de status van de simulatie bij te houden
1540      * @return true / false
1541      */
1542     public boolean isSimGestart(){
1543 
1544         return simIsGestart;
1545 
1546     }
1547 
1548     /**
1549      * Deze methiode wordt gebruikt om de status van de simulatie te weergeven als je wel of niet gepauzeerd is
1550      * @return hij geeft een true of false terug
1551      */
1552     public boolean isSimGepazeerd(){
1553 
1554         return simIsPauze;
1555     }
1556 
1557     //start & stop button
1558     /**
1559      * @author dareal
1560      * Deze inwedigge klasse met methode start of stop de simulatie. Het gebreurd met behulp van de iterator I. en het moet ook aan enkele
```

```
1561        * voorwaarde voldoen. ALs de simulatie gepauzeerd is dan gaat het niet door.
1562        */
1563       class StartLabelWissel implements ActionListener {
1564           public void actionPerformed(ActionEvent e) {
1565
1566               if(simIsPauze){
1567                   JOptionPane.showMessageDialog(frame, "Hervat eerst de sim, alvorens te stoppen.","Fout",
1568                           JOptionPane.ERROR_MESSAGE);
1569               }
1570               if(i == 0){
1571                   status.setText("Gestart");
1572                   simIsGestart = true;
1573                   zetLabelOpTrue();
1574                   start.setText("Simulatie Stoppen");
1575                   openPaneel();
1576                   hBaan.setVisible( true );
1577                   i++;
1578                   startTimer();
1579                   verwijder();
1580                   controller.startSimulatie(i);
1581                   zetLabelOpTrue();
1582               }
1583               else if(i == 1 && simIsPauze == false){
1584                   y = 0;
1585                   z = 0;
1586                   status.setText("Gestopt");
1587
1588                   simIsGestart = false;
1589                   isSimGestart();
1590                   TimertoString();
1591                   gemiddelStatistiek();
1592                   start.setText("Simulatie Starten");
1593                   i--;
1594                   stopTimer();
1595                   verwijder();
1596                   controller.zetAllStationNull();
1597
1598                   //zet de stations, baan en knoppen weer op invisible
1599                   s1_kn.setVisible( false );
1600                   s2_kn.setVisible( false );
1601                   s3_kn.setVisible( false );
1602                   s4_kn.setVisible( false );
1603                   s5_kn.setVisible( false );
1604                   s6_kn.setVisible( false );
1605                   s7_kn.setVisible( false );
1606                   s8_kn.setVisible( false );
1607
1608                   t1_kn.setVisible( false );
1609                   t2_kn.setVisible( false );
1610                   t3_kn.setVisible( false );
1611                   t4_kn.setVisible( false );
1612                   t5_kn.setVisible( false );
1613                   t6_kn.setVisible( false );
1614                   t7_kn.setVisible( false );
1615                   t8_kn.setVisible( false );
1616
1617                   s1_label.setVisible( false );
1618                   s2_label.setVisible( false );
1619                   s3_label.setVisible( false );
1620                   s4_label.setVisible( false );
1621                   s5_label.setVisible( false );
1622                   s6_label.setVisible( false );
1623                   s7_label.setVisible( false );
1624                   s8_label.setVisible( false );
1625                   hBaan.setVisible( false );
1626                   zelLabelOpFalse();
1627                   controller.startSimulatie(i);
1628                   zetLabelTextOpNull();
1629                   _station.clear();
1630                   //nieuw object van Controllerklasse
1631               }
```

```
1632            }
1633        }
1634
1635
1636        //hervat/pauzeer button
1637        /**
1638         * @author dareal
1639         *  Deze methode in de inwendigge klasse pauzeer of hervat de simulatuie d.m.v. een iterator genaamd y. Maar het voldoet wel aan een voorwaarde.
1640         * als de simulatie nog niet gestart is dan kan je deze methode niet uitvoeren
1641         */
1642        class HervatLabelWissel implements ActionListener {
1643            public void actionPerformed(ActionEvent e){
1644                System.out.println(reisManager.getStatus());
1645
1646                if(!(simIsGestart)){
1647                    JOptionPane.showMessageDialog(frame, "Start eerst de sim!","Fout",
1648                            JOptionPane.ERROR_MESSAGE);   }
1649
1650                if(y == 0 && simIsGestart ){
1651                    status.setText("Gepauzeerd");
1652
1653                    simIsPauze = true;
1654                    UserInterface.isGepauzeerd(simIsPauze);
1655                    isSimGepazeerd();
1656                    System.out.println(reisManager.getStatus());
1657                    y++;
1658                    controller.pauzeerSimulatie(y);
1659                    pauzeTimer();
1660                }
1661
1662                else if(y == 1){
1663                    status.setText("Gestart");
1664
1665                    simIsPauze = false;
1666                    UserInterface.isGepauzeerd(simIsPauze);
1667                    y--;
1668                    controller.pauzeerSimulatie(y);
1669                    timer.start();
1670                }
1671            }
1672        }
1673        //vernsel en normaal button
1674        /**
1675         * @author dareal
1676         * Deze methode in de inwendigge klasse versnel en normaliceer de simulatuie d.m.v een iterator genaamd z. Maar het moet wel aan een voorwaarde voldoen
1677         * betreft jey simulatie mnoet wel gestart zijn
1678         *
1679         */
1680        class VernselLabelWissel implements ActionListener {
1681            public void actionPerformed(ActionEvent e){
1682
1683                if(!(simIsGestart)){
1684                    JOptionPane.showMessageDialog(frame, "Start eerst de sim!","Fout",
1685                            JOptionPane.ERROR_MESSAGE);
1686                }
1687
1688
1689                if(z == 0 && simIsGestart){
1690                    status.setText("Versneld");
1691
1692                    z++;
1693                    controller.setVersnelSimulatie(z);
1694
1695
1696                }
1697                else if(z == 1){
1698                    status.setText("Gestart");
1699
1700                    z--;
1701                    controller.setVersnelSimulatie(z);
1702                }
```

```
1703            }
1704        }
1705
1706        /**
1707         * @author dareal
1708         *  Deze methode laad een moment opname. Maar voor dat hij dat doet moet er wel een data.dat bestand aanwezig zijn.
1709         */
1710        class MomentOpenen implements ActionListener {
1711            public void actionPerformed(ActionEvent arg0) {
1712                boolean exists = (new File("data.dat")).exists();
1713
1714                if(exists){
1715                if(i == 0){
1716                    status.setText("Gestart");
1717                        verwijder();
1718                        //start het simulatie
1719                        controller.startSimulatie(1);
1720                        controller.pauzeerSimulatie(1);
1721                        zetLabelOpTrue();
1722                        start.setText("Simulatie Stoppen");
1723                        y = 0;
1724                        z = 0;
1725                        i++;
1726                        simIsGestart = true;
1727                        UserInterface.isGestart(simIsGestart);
1728                        isSimGestart();
1729                        simIsPauze = false;
1730                        UserInterface.isGepauzeerd(simIsPauze);
1731                        isSimGepazeerd();
1732                        voegStationsToe(controller.stationGegevensOphalen());
1733                        voegTaxiToe(controller.taxiGegevensOphalen());
1734                        controller.voegObservToe(controller.taxiGegevensOphalen());
1735                        hBaan.setVisible( true );
1736                        startTimer();
1737                        zetLabelOpTrue();
1738
1739                        controller.pauzeerSimulatie(0);
1740                        //controller.startSimulatie(1);
1741                        System.out.println(" statuscheckdd"+ reisManager.getStatus());
1742                    }
1743                else {
1744
1745                    JOptionPane.showMessageDialog(frame, "De simulatie is al gestart.","Fout",
1746                            JOptionPane.ERROR_MESSAGE);
1747                    }
1748                }
1749                else{
1750                    JOptionPane.showMessageDialog(frame, "Bestand niet gevonden","Fout",
1751                            JOptionPane.ERROR_MESSAGE);
1752                    }
1753            }
1754        }
1755        //textvak acties!
1756        /**
1757         * @author dareal
1758         *  Deze inwendigge klasse methode voegt de aantal reiziger toe in de simulatie. Als de simulatie niet gestart is dam kan je dit niet uitvoeren
1759         * Verder moet het aan enkele voorwaardes voldoen wat in de if statements verwerk zijn. Hier kan je invoegen door op enter te drukken
1760         *
1761         */
1762        public class GetAantalReis implements ActionListener {
1763            public void actionPerformed(ActionEvent e){
1764
1765                if(!(simIsGestart)){
1766                    JOptionPane.showMessageDialog(frame, "Start eerst de sim!","Fout",
1767                            JOptionPane.ERROR_MESSAGE);
1768                }
1769
1770
1771                try{
1772                    String vertrekpunt = (String)vertrek.getSelectedItem();
1773                String bestemming = (String)aankomst.getSelectedItem();
```

```
1774            String aantalReis = aantal.getText();
1775            int aantalReiz = Integer.parseInt(aantalReis);
1776            if(aantalReiz < LIMIET_REIZIGER){
1777                if(i == 1){
1778                    if(!(vertrekpunt == null) && !(bestemming == null)){
1779                        if(!(vertrekpunt == bestemming)){
1780                            controller.voegReizigerToe(aantalReiz, vertrekpunt, bestemming);
1781                            JOptionPane.showMessageDialog(frame, "Reiziger is toegevoegd");
1782                            aantal.setText(null);
1783                            System.out.println(aantalReiz +  vertrekpunt + bestemming);
1784                            vertrek.setSelectedItem(null);
1785                            aankomst.setSelectedItem(null);
1786                            repaint();
1787                        }
1788                        else {
1789                            JOptionPane.showMessageDialog(frame, "vertrekpunt en bestemming zijn gelijk","Fout",
1790                                    JOptionPane.ERROR_MESSAGE);
1791                        }
1792                    }
1793                    else {
1794                        JOptionPane.showMessageDialog(frame, "vertrekpunt of bestemming is niet geslecteert","Fout",
1795                                JOptionPane.ERROR_MESSAGE);
1796                    }
1797                }
1798                else {
1799                    JOptionPane.showMessageDialog(frame, "Simulatie is nog niet gestart","Fout",
1800                            JOptionPane.ERROR_MESSAGE);
1801                }
1802            }
1803            else {
1804                JOptionPane.showMessageDialog(frame, "Limiet is overschreden","Fout",
1805                        JOptionPane.ERROR_MESSAGE);
1806            }
1807            }catch(NumberFormatException e1){
1808                JOptionPane.showMessageDialog(frame, "Vul een geldige waarde in.","Fout",
1809                        JOptionPane.ERROR_MESSAGE);
1810            }
1811
1812        }
1813    }
1814    /**
1815     * @author dareal
1816     *  Deze inwendigge klasse methode voegt de aantal reiziger toe in de simulatie. Als de simulatie niet gestart is dam kan je dit niet uitvoeren
1817     * Verder moet het aan enkele voorwaardes voldoen wat in de if statements verwerk zijn. Hier kan je invoegen door op het knop te drukken
1818     *
1819     */
1820    //Invoegen button acties
1821    class ReizigerInvoegen implements ActionListener {
1822        public void actionPerformed(ActionEvent e){
1823
1824            if(!(simIsGestart)){
1825                JOptionPane.showMessageDialog(frame, "Start eerst de sim!","Fout",
1826                        JOptionPane.ERROR_MESSAGE);
1827            }
1828
1829            try{
1830                String vertrekpunt = (String)vertrek.getSelectedItem();
1831            String bestemming = (String)aankomst.getSelectedItem();
1832            String aantalReis = aantal.getText();
1833            int aantalReiz = Integer.parseInt(aantalReis);
1834            if(aantalReiz < LIMIET_REIZIGER){
1835                if(i == 1){
1836                    if(!(vertrekpunt == null) && !(bestemming == null)){
1837                        if(!(vertrekpunt == bestemming)){
1838                            controller.voegReizigerToe(aantalReiz, vertrekpunt, bestemming);
1839                            JOptionPane.showMessageDialog(frame, "Reiziger is toegevoegd");
1840                            aantal.setText(null);
1841                            System.out.println(aantalReiz +  vertrekpunt + bestemming);
1842                            vertrek.setSelectedItem(null);
1843                            aankomst.setSelectedItem(null);
1844                            repaint();
```

```
1845                        }
1846                        else {
1847                            JOptionPane.showMessageDialog(frame, "vertrekpunt en bestemming zijn gelijk","Fout",
1848                                    JOptionPane.ERROR_MESSAGE);
1849                        }
1850                    }
1851                    else {
1852                        JOptionPane.showMessageDialog(frame, "vertrekpunt of bestemming is niet geslecteert","Fout",
1853                                JOptionPane.ERROR_MESSAGE);
1854                    }
1855                }
1856                else {
1857                    JOptionPane.showMessageDialog(frame, "Simulatie is nog niet gestart","Fout",
1858                            JOptionPane.ERROR_MESSAGE);
1859                }
1860            }
1861            else {
1862                JOptionPane.showMessageDialog(frame, "Limiet is overschreden","Fout",
1863                        JOptionPane.ERROR_MESSAGE);
1864            }
1865            }
1866            catch(NumberFormatException e1){
1867                JOptionPane.showMessageDialog(frame, "Vul een geldige waarde in.","Fout",
1868                        JOptionPane.ERROR_MESSAGE);
1869            }
1870
1871        }
1872    }
1873    /**
1874     * @author dareal
1875     * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1876     *
1877     */
1878    class Taxi_1 implements ActionListener {
1879        public void actionPerformed(ActionEvent e){
1880            new TaxiStats(controller.getPositie(1), controller.getReizInTaxi(1), controller.getReisTijd(1), 1);
1881        }
1882    }
1883    /**
1884     * @author dareal
1885     * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1886     *
1887     */
1888    class Taxi_2 implements ActionListener {
1889        public void actionPerformed(ActionEvent e){
1890            //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(2), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1891            new TaxiStats(controller.getPositie(2), controller.getReizInTaxi(2), controller.getReisTijd(2), 2);
1892        }
1893    }
1894    /**
1895     * @author dareal
1896     * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1897     *
1898     */
1899    class Taxi_3 implements ActionListener {
1900        public void actionPerformed(ActionEvent e){
1901            //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(3), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1902            new TaxiStats(controller.getPositie(3), controller.getReizInTaxi(3), controller.getReisTijd(3), 3);
1903        }
1904    }
1905    /**
1906     * @author dareal
1907     * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1908     *
1909     */
1910    class Taxi_4 implements ActionListener {
1911        public void actionPerformed(ActionEvent e){
1912            //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(4), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1913            new TaxiStats(controller.getPositie(4), controller.getReizInTaxi(4), controller.getReisTijd(4), 4);
1914        }
1915    }
```

```
1916      /**
1917       * @author dareal
1918       * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1919       *
1920       */
1921      class Taxi_5 implements ActionListener {
1922          public void actionPerformed(ActionEvent e){
1923              //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(5), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1924              new TaxiStats(controller.getPositie(5), controller.getReizInTaxi(5), controller.getReisTijd(5), 5);
1925          }
1926      }
1927      /**
1928       * @author dareal
1929       * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1930       *
1931       */
1932      class Taxi_6 implements ActionListener {
1933          public void actionPerformed(ActionEvent e){
1934              //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(6), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1935              new TaxiStats(controller.getPositie(6), controller.getReizInTaxi(6), controller.getReisTijd(6), 6);
1936          }
1937      }
1938      /**
1939       * @author dareal
1940       * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1941       *
1942       */
1943      class Taxi_7 implements ActionListener {
1944          public void actionPerformed(ActionEvent e){
1945              //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(7), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1946              new TaxiStats(controller.getPositie(7), controller.getReizInTaxi(7), controller.getReisTijd(7), 7);
1947          }
1948      }
1949      /**
1950       * @author dareal
1951       * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1952       *
1953       */
1954      class Taxi_8 implements ActionListener {
1955          public void actionPerformed(ActionEvent e){
1956              //new TaxiStats(controller.getPositie(), controller.getReizInTaxi(8), controller.getWachtTijd(), controller.getReisTijd(), controller.getGemBezet(), controller.getGemWachtTijd(), control
1957              new TaxiStats(controller.getPositie(8), controller.getReizInTaxi(8), controller.getReisTijd(8), 8);
1958          }
1959      }
1960      /**
1961       * @author dareal
1962       * Deze methode roept vult de controctor van de taxistats klasse(taxiu statustiek) met de juiste gegevens.
1963       *
1964       */
1965      class Amstel implements ActionListener {
1966          public void actionPerformed(ActionEvent a ){
1967              new StationStats(controller.getNaam(1), controller.getA_reizigers(1), controller.getA_taxis(1));
1968          }
1969      }
1970
1971
1972      /**
1973       * @author dareal
1974       *
1975       * Hier wordt bepaald welke statistiek aangeroepen dient te worden zodat het klopt volgens de gegevens
1976       * Dit heeft de maken met de statustieken van de station. Het wordt geactiveerd als je op een station knop drukt. Wat dan meegegeven wordt
1977       * is dan de aantal station die toegevoegd is in de somulatie.
1978       */
1979      //Dit is het versie onafhangkelijk van de algoritme
1980
1981      class Sloterdijk implements ActionListener {
1982          public void actionPerformed(ActionEvent a ){
1983
1984              switch(aantalStation){
1985              case 5:
1986                  new StationStats(controller.getNaam(1), controller.getA_reizigers(1), controller.getA_taxis(1));
```

```
1987                break;
1988            case 3:
1989                new StationStats(controller.getNaam(3), controller.getA_reizigers(3), controller.getA_taxis(3));
1990                break;
1991            default:
1992                new StationStats(controller.getNaam(2), controller.getA_reizigers(2), controller.getA_taxis(2));
1993            break;
1994            }
1995        }
1996    }
1997    class AmsterdamCS implements ActionListener {
1998        public void actionPerformed(ActionEvent a ){
1999            switch(aantalStation){
2000            case 6:
2001                new StationStats(controller.getNaam(2), controller.getA_reizigers(2), controller.getA_taxis(2));
2002                break;
2003            case 4:
2004                new StationStats(controller.getNaam(2), controller.getA_reizigers(2), controller.getA_taxis(2));
2005                break;
2006            default:
2007                new StationStats(controller.getNaam(3), controller.getA_reizigers(3), controller.getA_taxis(3));
2008            //baanManager.getStations();
2009            break;
2010            }
2011
2012        }
2013    }
2014    class Lelylaan implements ActionListener {
2015        public void actionPerformed(ActionEvent a ){
2016            switch(aantalStation){
2017            case 6:
2018                new StationStats(controller.getNaam(3), controller.getA_reizigers(3), controller.getA_taxis(3));
2019                break;
2020            case 5:
2021                new StationStats(controller.getNaam(2), controller.getA_reizigers(2), controller.getA_taxis(2));
2022                break;
2023            default:
2024                new StationStats(controller.getNaam(4), controller.getA_reizigers(4), controller.getA_taxis(4));
2025            break;
2026            }
2027        }
2028    }
2029
2030    class Muiderpoort implements ActionListener {
2031        public void actionPerformed(ActionEvent a ){
2032            switch(aantalStation){
2033            case 6:
2034                new StationStats(controller.getNaam(4), controller.getA_reizigers(4), controller.getA_taxis(4));
2035                break;
2036            case 5:
2037                new StationStats(controller.getNaam(3), controller.getA_reizigers(3), controller.getA_taxis(3));
2038                break;
2039            case 4:
2040                new StationStats(controller.getNaam(3), controller.getA_reizigers(3), controller.getA_taxis(3));
2041                break;
2042            case 3:
2043                new StationStats(controller.getNaam(1), controller.getA_reizigers(1), controller.getA_taxis(1));
2044                break;
2045            case 2:
2046                new StationStats(controller.getNaam(2), controller.getA_reizigers(2), controller.getA_taxis(2));
2047                break;
2048            default:
2049                new StationStats(controller.getNaam(5), controller.getA_reizigers(5), controller.getA_taxis(5));
2050            break;
2051
2052            }
2053        }
2054    }
2055    class ZuidWTC implements ActionListener {
2056        public void actionPerformed(ActionEvent a ){
2057            new StationStats(controller.getNaam(6), controller.getA_reizigers(6), controller.getA_taxis(6));
```

```
2058          }
2059      }
2060      class Rai implements ActionListener {
2061          public void actionPerformed(ActionEvent a ){
2062              switch(aantalStation){
2063              case 7:
2064                  new StationStats(controller.getNaam(6), controller.getA_reizigers(6), controller.getA_taxis(6));
2065                  break;
2066              case 6:
2067                  new StationStats(controller.getNaam(5), controller.getA_reizigers(5), controller.getA_taxis(5));
2068                  break;
2069              case 5:
2070                  new StationStats(controller.getNaam(4), controller.getA_reizigers(4), controller.getA_taxis(4));
2071                  break;
2072              case 4:
2073                  new StationStats(controller.getNaam(4), controller.getA_reizigers(4), controller.getA_taxis(4));
2074                  break;
2075              case 3:
2076                  new StationStats(controller.getNaam(3), controller.getA_reizigers(3), controller.getA_taxis(3));
2077                  break;
2078              default:
2079                  new StationStats(controller.getNaam(7), controller.getA_reizigers(7), controller.getA_taxis(7));
2080              break;
2081              }
2082          }
2083      }
2084      class Duivendrecht implements ActionListener {
2085          public void actionPerformed(ActionEvent a ){
2086              switch(aantalStation){
2087              case 7:
2088                  new StationStats(controller.getNaam(7), controller.getA_reizigers(7), controller.getA_taxis(7));
2089                  break;
2090              case 6:
2091                  new StationStats(controller.getNaam(6), controller.getA_reizigers(6), controller.getA_taxis(6));
2092                  break;
2093              case 5:
2094                  new StationStats(controller.getNaam(5), controller.getA_reizigers(5), controller.getA_taxis(5));
2095                  break;
2096              default:
2097                  new StationStats(controller.getNaam(8), controller.getA_reizigers(8), controller.getA_taxis(8));
2098              break;
2099              }                                           //ALs ke 7 station kiest dna moet je hier 7 invullen
2100          }
2101      }
2102
2103
2104      //inwendige klasse voor de timer!
2105      class ClockListener implements ActionListener {
2106          public void actionPerformed(ActionEvent e) {
2107              ++sec;//telt sec op
2108              if( sec == 60){
2109                  min++;//telt min op
2110                  sec = 0;//zet sec op 0
2111              }
2112              if(min == 60){
2113                  uur++;
2114                  min = 0;
2115              }
2116              tijdVeld.setVisible(true);
2117              //System.out.println("" + uur + ":" + min + ":" + sec);
2118              tijdVeld.setText("" + uur + ":" + min + ":" + sec);
2119
2120          }
2121
2122      }
2123      /**
2124       * @author dareal
2125       *  Hier wordt de inwendigge klasse aangeroepen die alleen de optionPane klasse aanroept
2126       */
2127      @SuppressWarnings("serial")
2128      class GemmidelstatistiekScherm extends JOptionPane {
```

```
2129
2130            /**
2131             * De optionpane wordt opgeroepen
2132             */
2133            GemmidelstatistiekScherm() {
2134
2135            }
2136        }
2137
2138
2139        //teken het achtergrondplaatje
2140        public void paintComponent( Graphics g) {
2141            super.paintComponent(g);
2142            achtergrond.paintIcon( this, g, 0, 0);
2143        }
2144
2145        @Override
2146        public void update(Observable o, Object arg) {
2147            Taxi t = (Taxi) o;
2148            teller++;
2149            if(teller == 5){
2150                teller = 0;
2151            }
2152            setLocation(t.getID(), new Double(t.getAbsolutePositie()/12.5).intValue(), new Double(t.getZijPositie()/12.5).intValue(),t.opStation(),t.naarStation(), t.vanStation(), teller, reisManager.ge
2153            System.out.println("-----------------------------------------------");
2154            System.out.println("geupdate");
2155            System.out.println("Absolute positie: " + t.getAbsolutePositie());
2156            System.out.println("Taxi nummer: " + t.getID());
2157            System.out.println("Zij positie: "+t.getZijPositie());
2158            System.out.println("Op station? " + t.opStation());
2159            System.out.println("naar station? " + t.naarStation());
2160            System.out.println("van station? " + t.vanStation());
2161            System.out.println(teller);
2162            System.out.println(reisManager.getVersnelFactor());
2163            System.out.println("-----------------------------------------------");
2164        }
2165    }
2166
```

InstellingScherm.java

```
1   /*----------------------------*/
2   /* This Program was made by:  */
3   /* Isabelle Hanraads          */
4   /* Any kind of fraude will be  */
5   /* seen as a serious crime and */
6   /* will be punished by a money */
7   /* penalty of 25,-            */
8   /*____________________________*/
9
10  import java.awt.BorderLayout;
11  import java.awt.Container;
12  import java.awt.Frame;
13  import java.awt.Graphics;
14  import java.awt.Image;
15  import java.awt.MediaTracker;
16  import java.awt.TextField;
17  import java.awt.Toolkit;
18  import java.awt.event.ActionEvent;
19  import java.awt.event.WindowAdapter;
20  import java.awt.event.WindowEvent;
21  import java.awt.*;
22  import java.awt.event.*;
23  import java.net.URL;
24
25  import javax.swing.*;
26
27  import javax.swing.ImageIcon;
28  import javax.swing.JButton;
29  import javax.swing.JFrame;
30  import javax.swing.JLabel;
31  import javax.swing.JOptionPane;
32  import javax.swing.JPanel;
33  import javax.swing.JTextField;
34
35
36  //Frame-klasse voor het Instellingen pop-up venster
37  /**
38   * @author dareal
39   * Deze klasse tekent een paneel in een frame en wordt gebruikt voor de taxi instellingen.
40   * Je kan de snelheid van de taxi instellen en de reixigers instaptijd.
41   * De snelheid moet tussen 60 en 300 km/u liggen en de instaptijd mag vanaf 2 seconde en
42   * hoger zijn.
43   *
44   */
45  public class InstellingScherm extends Frame {
46      private JFrame frame;
47      private JPanel instellingscherm;
48      private JTextField snelheid,wachttijd;
49      private ImageIcon taxi;
50      private JButton okay, annuleren;
51      private Controller controller;
52
53
54      public InstellingScherm(){
55          taxi = new ImageIcon("src/taxi.gif");
56          frame = new JFrame("Baaninstellingen");
57          controller = new Controller();
58          instellingscherm = new JPanel(){
59              protected void paintComponent(Graphics g)
60              {
61                  /**
62                   * Teken het plaatje op het volledig scherm
63                   **/
64                  g.drawImage(taxi.getImage(), 0, 0, null);
65
66                  super.paintComponent(g);
67              }
68          };
69          addWindowListener( new Sluiten());
```

```
70
71          /**
72           * knoppen toevoegen
73           * */
74          okay = new JButton("OK");
75          okay.setSize(53,30);
76          okay.setLocation(110,230);
77          okay.addActionListener(new Ok());
78
79          annuleren = new JButton("Annuleren");
80          annuleren.setSize(100,30);
81          annuleren.setLocation(180,230);
82          annuleren.addActionListener(new Annuleren());
83
84          /**
85           * Textfield aanmaken
86           * */
87          snelheid = new JTextField();
88          snelheid.setSize(120,25);
89          snelheid.setLocation(20, 100);
90
91          wachttijd = new JTextField();
92          wachttijd.setSize(120,25);
93          wachttijd.setLocation(20,180);
94
95          /**
96           * Objecten toevoegen in de frame
97           * */
98          frame.add(snelheid);
99          frame.add(wachttijd);
100         frame.add(okay);
101         frame.add(snelheid);
102         frame.add(wachttijd);
103         frame.add(annuleren);
104
105         instellingscherm.setOpaque( false );
106         frame.getContentPane().add(instellingscherm, BorderLayout.CENTER);
107         frame.setSize( 300, 300);
108         frame.setVisible( true );
109     }
110
111     /**
112      * inwendige klasse voor het sluiten van de scherm
113      * */
114     class Annuleren implements ActionListener {
115         public void actionPerformed(ActionEvent e) {
116             frame.dispose();
117
118         }
119
120     }
121     /**
122      * inwendige klasse voor als je op het ok knop druk om de gegevens toe te voegen
123      *
124      * */
125     class Ok implements ActionListener {
126         public void actionPerformed(ActionEvent e) {
127             try{
128                 String snelheid1 = snelheid.getText();
129                 int snelHeid = Integer.parseInt(snelheid1);
130                 if(snelHeid >= 60 && snelHeid <= 300){
131
132                     controller.setSnelheid(snelHeid);
133                 }
134                 else {
135                     JOptionPane.showMessageDialog(frame, " De betreffende snelheid " + snelheid1 + " is geen juiste invoer.\n Het moet tussen 60 en 300 km/u liggen","Fout",
136                             JOptionPane.ERROR_MESSAGE);
137                 }
138             }
139             catch(NumberFormatException e1){
140             }
```

```
141             frame.dispose();
142             try{
143                 String wachttijd1 = wachttijd.getText();
144                 int wachttijd = Integer.parseInt(wachttijd1);
145                 if(wachttijd >= 2){
146                     controller.setWachttijd(wachttijd);
147                 }
148                 else {
149                     JOptionPane.showMessageDialog(frame, " De betreffende limiet, " + wachttijd1 + " is geen juiste invoer.\n U dient een getal groter of gelijk aan 2 in te voeren","Fout",
150                             JOptionPane.ERROR_MESSAGE);
151                 }
152             }
153             catch(NumberFormatException e1){
154             }
155 
156             frame.dispose();
157         }
158 
159     }
160     /**
161      * inwendige klasse voor het sluiten van het scherm
162      * */
163     private class Sluiten extends WindowAdapter {
164         public void windowClosing( WindowEvent e){
165             frame.dispose();
166 
167         }
168     }
169 
170 }
171 
172 
```

Controller.java

```
1   import java.util.ArrayList;
2   import java.util.Observer;
3   import java.util.StringTokenizer;
4
5   import javax.swing.JComboBox;
6   import javax.swing.JTextField;
7
8   import org.tigam.railcab.algoritme.*;
9
10
11  /**
12   * @author dareal
13   *Deze klasse verzorgd het communicatie tussen het algoritme en de UserInterface
14   */
15  public class Controller {
16      private int meters, wissel_sp, km, pixel_x, pixel_y;
17      private double pixels, afst_stations, bocht, station;
18      private String statistieken, ReizigersOpstation;
19
20      private UserInterface userInterface;
21      private ArrayList<String> reiziger;
22      private BaanManager baanManager;
23      private ReisManager reisManager;
24      private Simulatie simulatie;
25      private TaxiStats taxiStats;
26      private TekenSpoor tekenSpoor;
27      private Taxi taxi;
28
29      //stations attributten
30      private String station1, station2, station3, station4, station5, station6,
31              station7, station8;
32
33      //attibuten voor taxistats
34      private int aantal, r_inT, wT, rT, gB, g_wT, g_rT, aantalTaxi = 0;
35
36      //attributen voor stationstats
37
38      private int r_station1, r_station2, r_station3, r_station4, r_station5,
39              r_station6, r_station7, r_station8, iterator;
40      private String vertrek, bestemming, pos;
41      private Paneel paneel;
42
43      public Controller() {
44          simulatie = Simulatie.getInstantie();
45          baanManager = simulatie.getBaanManager();
46          reisManager = simulatie.getReisManager();
47          tekenSpoor = new TekenSpoor();
48      }
49
50      public Controller(Paneel paneel) {
51
52          //1 pixel = 12.5 meters
53
54          this.wissel_sp = 50; //meters
55          this.meters = 24000; //meters
56          this.pixels = 12.5; //m/pixel
57          this.afst_stations = 62.5 * 13; //pixels
58          this.bocht = 13 * 62.5; //pixels
59          this.station = 10 * 62.5; //pixels
60          simulatie = Simulatie.getInstantie();
61          baanManager = simulatie.getBaanManager();
62          reisManager = simulatie.getReisManager();
63          this.paneel = paneel;
64
65      }
66
67      /**
68       * Deze methode voegd de reizigers toe aan de algoritme d.m.v. een forloop
69       * */
```

```
70      public void voegReizigerToe(int aantal, String vertrek, String bestemming) {
71          for (int i = 0; i < aantal; i++) {
72              reisManager.addReiziger(baanManager.getStation(vertrek),
73                      baanManager.getStation(bestemming));
74          }
75      }
76
77      /**
78       * Deze methode stelt de snelheid in voor de taxi. Hij zet de km om naar meters per seconde
79       * */
80      public void setSnelheid(double snelheid) {
81          System.out.println("Ingevoerde snelheid: " + snelheid + "KM/h\n");
82          snelheid = snelheid / 3.6;
83          try {
84              baanManager.setTaxiSnelheid((int) snelheid);
85              //doorsturen naar ReisManager
86          } catch (Exception e) {
87              System.out.println(" de ingevoerde snelheid " + snelheid
88                      + " komt niet aan");
89          }
90      }
91
92      /**
93       * Deze methode wijzig de reizigers taxi instaptijd.
94       * */
95      public void setWachttijd(int wachttijd) {
96          System.out.println(" de wachttijd is" + wachttijd);
97          wachttijd = wachttijd * 1000;
98          System.out.println(wachttijd);
99          try {
100             baanManager.setWachtTijdStation(wachttijd);
101             //doorsturen naar ReisManager
102         } catch (Exception e) {
103             System.out.println(" de wachtijd  " + wachttijd + " komt niet aan");
104         }
105     }
106
107     /**
108      * Deze methode bereken de gemiddelde statistieken en geeft het terug in minuten en secondes
109      * */
110     public String getGemStatistieken() {
111         int gemReistijdSec = (int) (simulatie.gemiddeldeReisTijd() / 1000), gemReistijdMin = (int) (simulatie
112                 .gemiddeldeReisTijd() / 1000 / 60), gemiddelWachtijdMin = (int) (simulatie
113                 .gemiddeldeWachtTijd() / 1000 / 60), gemiddelWachtijdSec = (int) (simulatie
114                 .gemiddeldeWachtTijd() / 1000);
115         long gemiddelBezetting = simulatie.gemiddeldeTaxiBezetting();
116         statistieken = "Gemiddelde TaxiBezetting: " + gemiddelBezetting + " %"
117                 + "\n" + "Gemiddelde WachtTijd: " + gemiddelWachtijdMin + "."
118                 + secondeCheck(gemiddelWachtijdSec) + " Minuten" + "\n"
119                 + "Gemiddelde ReisTijd: " + gemReistijdMin + "."
120                 + secondeCheck(gemReistijdSec) + " Minuten";
121
122         return statistieken;
123     }
124
125     public int versnelTimer() {
126         int reken = 1000 / ReisManager.getVersnelFactor();
127         return reken;
128     }
129
130     /**
131      * Deze methode versnelt de simulatie
132      *
133      * */
134     public void setVersnelSimulatie(int i) {
135         if (i == 1) {
136             simulatie.versnel();
137             versnelTimer();
138             paneel.VersnelOfNormaliceerTimer(versnelTimer());
139         } else if (i == 0) {
140             simulatie.versnel();
```

```
141             paneel.VersnelOfNormaliceerTimer(1000);
142         }
143     }
144
145     /**
146      * Deze methode start de simulatie op vanuit de algoritme
147      * */
148     public void startSimulatie(int i) {
149         if (i == 1) {
150
151             baanManager = simulatie.getBaanManager();
152             reisManager = simulatie.getReisManager();
153             simulatie.start();
154             System.out.println("simulatie start op");
155         } else {
156             simulatie.stop();
157             System.out.println("simulatie stopt");
158
159         }
160     }
161
162     /**
163      * Deze methode pauzeer of hervat de simulatie. Het wordt in de paneel klasse aangeroepen.
164      * */
165     public void pauzeerSimulatie(int i) {
166         if (i == 1) {
167             simulatie.pauzeer();
168         } else {
169             simulatie.hervat();
170         }
171     }
172
173     /**
174      * Deze methode geeft het positie terug van een taxi. Een taxi is dan op een spoor of op een station
175      * */
176     public String getPositie(int i) {
177         if (baanManager.getTaxi(i).getStation() == null) {
178             return "Op spoor";
179         } else {
180             return "Op station";
181
182         }
183
184     }
185
186     /**
187      * Deze methode geeft het aantal reiziger terug die in een taxi bevinden
188      * */
189     public int getReizInTaxi(int i) {
190         try {
191             return baanManager.getTaxi(i).getAantalReizigers();
192         } catch (Exception e) {
193             return 0;
194         }
195
196     }
197
198     /**
199      * Deze methode zorgd ervoor dat de secondes aangegeven wordt als de standaard. dus van 1 tot 60
200      * */
201     public int secondeCheck(int sec) {
202         for (int i = 0; sec > 60; i++) {
203             sec = sec - 60;
204         }
205         return sec;
206     }
207
208     /**
209      * Deze methode geeft de reistijd terug van een taxi. Tevens mmaakt hij gebruikt van een berekening die de milisecondes omzet
210      * in secondes en minuten.
211      * */
```

```
212     public String getReisTijd(int i) {
213
214         try {
215             int minuut = (int) (baanManager.getTaxi(i).getReis().getReisTijd() / 1000 / 60);
216             int seconde = (int) (baanManager.getTaxi(i).getReis().getReisTijd() / 1000);
217             return minuut + "." + secondeCheck(seconde);
218         } catch (Exception e) {
219             return "0";
220         }
221     }
222
223     /**
224      * Deze methode geeft een station naam terug
225      * */
226
227     public String getNaam(int i) {
228         try {
229             return baanManager.getStation(i).getNaam();
230         } catch (Exception e) {
231             return "Geen Station";
232         }
233     }
234
235     /**
236      * Geeft de aantal reizigers op station
237      * */
238     public int getA_reizigers(int i) {
239         try {
240             return baanManager.getStation(i).getAantalReizigers();
241         } catch (Exception e) {
242             return 0;
243         }
244
245     }
246
247     /**
248      * Geeft de aantal taxi op een station terug.
249      * */
250     public int getA_taxis(int i) {
251         try {
252             return baanManager.getStation(i).getAantalTaxis();
253         } catch (Exception e) {
254             return 0;
255         }
256
257     }
258
259     public int getStaat(int a) {
260         return a;
261         // TODO Auto-generated method stub
262
263     }
264
265     /**
266      * Deze methode geeft alle station atributten een verwijzing naar een station naam.
267      * */
268     public void setStation(String nm) {
269
270         if (station1 == null) {
271             station1 = nm;
272         } else if (station2 == null) {
273             station2 = nm;
274         } else if (station3 == null) {
275             station3 = nm;
276         } else if (station4 == null) {
277             station4 = nm;
278         } else if (station5 == null) {
279             station5 = nm;
280         } else if (station6 == null) {
281             station6 = nm;
282         } else if (station7 == null) {
```

```
283             station7 = nm;
284         } else if (station8 == null) {
285             station8 = nm;
286         }
287 
288     }
289 
290     /**
291      * Deze methode voegt de aantal station in de algritme toe
292      * */
293     public void setStations(int aantalStation) {
294         baanManager.maakStations(aantalStation);
295         System.out
296                 .println("aantalStatation die naar baanmanager wordt gestuurd"
297                         + aantalStation);
298     }
299 
300     /**
301      * Deze methode voegt de aantal taxis toe in de algoritme. en teken ook de taxis op het UI.
302      * */
303     public void setTaxi(int aantalTaxi) {
304         System.out.println(" aantalTaxi In controller" + aantalTaxi);
305         baanManager.maakTaxis(aantalTaxi, paneel);
306 
307         for (int b = 0; b < baanManager.getTaxis().size(); b++) {
308             paneel.setInitLoc(baanManager.getTaxis().get(b)
309                     .getAbsolutePositie(), baanManager.getTaxis().get(b)
310                     .getZijPositie(), b + 1);
311         }
312     }
313 
314     /**
315      * Deze methode pas observable toe op de aantal taxis die toegevoegd worden vanuit de momentopname
316      *
317      * */
318     public void voegObservToe(int aantalTaxi) {
319         ArrayList<Taxi> taxis = baanManager.getTaxis();
320         for (Taxi t : taxis) {
321             t.addObserver(paneel);
322         }
323 
324     }
325 
326     /**
327      * Deze methode wordt aangeroepen om de taxi gegevens op te halen voor de UI.
328      * Hij geeft de aantal taxis aan die in data.dat opgeslagen is zodat het in de UI teogevoegd
329      * kan worden.
330      **/
331     public int taxiGegevensOphalen() {
332         simulatie.laden("data.dat");
333         baanManager = Simulatie.getInstantie().getBaanManager();
334         reisManager = Simulatie.getInstantie().getReisManager();
335 
336         return baanManager.getAantalTaxis();
337     }
338 
339     /**
340      * Deze methode wordt aangeroepen om de stations gegevens op te halen voor de UI.
341      * Hij geeft de aantal stations die in data.dat opgeslagen is zodat het in de UI teogevoegd
342      * kan worden.
343      **/
344     public int stationGegevensOphalen() {
345         simulatie.laden("data.dat");
346         baanManager = Simulatie.getInstantie().getBaanManager();
347         reisManager = Simulatie.getInstantie().getReisManager();
348 
349         return baanManager.getStations().length;
350     }
351 
352     /*Deze methode wordt gebruikt om een momentOpname op te slaan. De UI klasse roept dit aan.
353      **/
```

```
354       public void momentOpnameOpslaan() {
355           simulatie.opslaan("data.dat");
356       }
357
358       /**
359        *Deze methode geeft alle attributen een null/0 verwijzing. Dit wordt pas aangeroepen als je de simulatie stop.
360        *En is noodzakelijk zodat je de simulatie in een lege status kan opstarten vanuit de UI.
361        * */
362       public void zetAllStationNull() {
363           station1 = null;
364           station2 = null;
365           station3 = null;
366           station4 = null;
367           station5 = null;
368           station6 = null;
369           station7 = null;
370           station8 = null;
371           r_station1 = 0;
372           r_station2 = 0;
373           r_station3 = 0;
374           r_station4 = 0;
375           r_station5 = 0;
376           r_station6 = 0;
377           r_station7 = 0;
378           r_station8 = 0;
379       }
380
381   }
382
```

ArrayListComboBoxModel.java

```
1   import java.awt.BorderLayout;
2   import java.util.ArrayList;
3   import java.util.Collection;
4
5   import javax.swing.AbstractListModel;
6   import javax.swing.ComboBoxModel;
7   import javax.swing.JComboBox;
8   import javax.swing.JFrame;
9
10  /**
11   * @author dareal
12   * Deze klasse is een abstracte klasse die alle methodes van de AbstractListModel implmenenteert
13   * Dit wordt voor de paneel klasse gebruikt om een array van de stations te maken.
14   */
15  public class ArrayListComboBoxModel extends AbstractListModel implements ComboBoxModel {
16    private Object selectedItem;
17
18    private ArrayList anArrayList;
19
20    public ArrayListComboBoxModel(ArrayList arrayList) {
21      anArrayList = arrayList;
22    }
23
24    public Object getSelectedItem() {
25      return selectedItem;
26    }
27
28    public void setSelectedItem(Object newValue) {
29      selectedItem = newValue;
30    }
31
32    public int getSize() {
33      return anArrayList.size();
34    }
35
36    public Object getElementAt(int i) {
37      return anArrayList.get(i);
38    }
39  }
40
```