**Code Base**

for

**Train Controller (NSECS-TNC)**

**Prepared by**

BreathlessBovine: Ben Tomasulo

**Organization**

ECE/COE 1186: Software Engineering

**Date**

4/25/2013

**Class PowJar:**

package TNM;

/\*\*

\* @class PowJar

\*

\* @date 04/24/2013

\*

\* @author Ben Tomasulo

\*/

/\*\*

\* This file contains the container class which holds the

\* power response of a Train Controller PLC

\*

\*/

public class PowJar {

public double jPower;

public double jUk;

public double jErr;

public boolean jEBrake;

public boolean jSBrake;

public PowJar(double jPower, double jUk, double jErr, boolean jEBrake, boolean jSBrake) {

this.jPower=jPower;

this.jUk=jUk;

this.jErr=jErr;

this.jEBrake=jEBrake;

this.jSBrake=jSBrake;

}

}

**Class ResponseUI:**

package TNM;

/\*\*

\* @class ResponseUI

\*

\* @date 04/23/2013

\*

\* @author Ben Tomasulo

\*/

/\*\*

\* This file contains the main container class which holds the response between a

\* given train's controller and the UI after each timeTick.

\*/

public class ResponseUI {

public double cvo;

public boolean cvoT;

public double currSetpoint;

public double slo;

public boolean sloT;

public double autO;

public boolean autOT;

public int tbnO;

public boolean tbnOT;

public double timeO;

public boolean timeOT;

public int trainEFoT;

public int signalPFoT;

public int brakeFoT;

public int brokenRoT;

public int trackCFoT;

public int powerFoT;

public int sBrakeOT;

public int eBrakeOT;

public double cit;

public boolean paused;

public ResponseUI(double cvo, boolean cvoT, double currSetpoint, double slo, boolean sloT,

double autO, boolean autOT, int tbnO, boolean tbnOT, double timeO, boolean timeOT,

int trainEFoT, int signalPFoT, int brakeFoT, int brokenRoT, int trackCFoT, int powerFoT,

int sBrakeOT, int eBrakeOT, double cit, boolean paused) {

this.cvo=cvo;

this.cvoT=cvoT;

this.currSetpoint=currSetpoint;

this.slo=slo;

this.sloT=sloT;

this.autO=autO;

this.autOT=autOT;

this.tbnO=tbnO;

this.tbnOT=tbnOT;

this.timeO=timeO;

this.timeOT=timeOT;

this.trainEFoT=trainEFoT;

this.signalPFoT=signalPFoT;

this.brakeFoT=brakeFoT;

this.brokenRoT=brokenRoT;

this.trackCFoT=trackCFoT;

this.powerFoT=powerFoT;

this.sBrakeOT=sBrakeOT;

this.eBrakeOT=eBrakeOT;

this.cit=cit;

this.paused=paused;

}

}

**Class TNC\_Test:**

package TNM;

import TKM.\*;

/\*\*

\* NSECS-TNC TEST INTERFACE

\*

\* @author (Ben Tomasulo)

\* @version (4/24/2013)

\*/

public class TNC\_Test

{

public static TNC\_UI tncUI;

public static TrainController testTNC;

public static Block negout = new Block(-3, "A", "I", 100.0, 3.0, 60.0, false, false, true, false, false, "You found the secret stop, exit now to collect 100 rupies!", false, false, false);

public static Block negone = new Block(-1, "B", "II", 100.0, 3.0, 55.0, false, false, false, false, false, "", false, false, false);

public static void main(String [] args)

{

testTNC=new TrainController();

tncUI=new TNC\_UI(true);

for(int i=28800; i<86400; i++)

{

ResponseUI tncData=tncUI.updateUI(testTNC);

testTNC.updateUI(tncData);

testTNC.timeTick(i,-1, 1, negone, negout, false, false, false, 100, 100, false, true, negone.transponderMessage);

long t0,t1;

t0=System.currentTimeMillis();

do{

t1=System.currentTimeMillis();

}

while (t1-t0<100);

}

}

}

**Class TrainController:**

package TNM;

import TKM.\*;

/\*\*

\* NSECS-TNC INDIVIDUAL CONTROL MODULE

\* REQUIRES EXTERNAL CLASS TO ACTIVATE

\* EITHER TNM or TNC-UI

\*

\* @author Ben Tomasulo

\* @version 4/22/2013

\*/

public class TrainController

{

   // instance variables

   public double power;

   public boolean sBrake;

   public boolean eBrake;

   public boolean lights;

   public boolean doors;

   public double tTemp;

   public Block thisBlock;

   public Block oldBlock;

   public double time;

   public double setpoint;

   public double autspeed;

   public double slimit;

   public double safespeed;

   public double currspeed;

   public final double maxpower=120000;

   public boolean signalFail;

   public boolean brakeFail;

   public boolean engineFail;

   public boolean railFail;

   public boolean trackFail;

   public boolean powerFail;

   public int stationState;

   public double dtime;

   public String currAnnoun;

   public double tperiod;

   public final double Kp=800;

   public final double Ki=200;

   public double ukprev;

   public double prerr;

   public boolean opstate;

   public String nextName;

   public PowJar powone;

   public PowJar powtwo;

   public PowJar powthree;

   public double spone;

   public double sptwo;

   public double spthree;

   public int checkid;

   private ResponseUI overR;

   // create test blocks

   Block negout = new Block(-3, "A", "I", 100.0, 3.0, 60.0, false, false, true, false, false, "You found the secret stop, exit now to collect 100 rupies!", false, false, false);

   Block negone = new Block(-1, "B", "II", 100.0, 3.0, 55.0, false, false, false, false, false, "", false, false, false);

   Block negtwo = new Block(-2, "C", "III", 100.0, 3.0, 55.0, false, true, false, true, false, "-2", false, false, false);

   /\*\*

    \* Constructor for objects of class TrainController

    \*/

   public TrainController()

   {

       // initialise instance variables

       tTemp=22;

       sBrake=true;

       eBrake=false;

       lights=false;

       doors=true;

       thisBlock=negone;

       oldBlock=thisBlock;

       time=-1;

       setpoint=0;

       autspeed=100;

       slimit=-1;

       setLights();

       safespeed=0;

       currspeed=0;

       signalFail=false;

       brakeFail=false;

       engineFail=false;

       railFail=thisBlock.brokenRailFailure;

       trackFail=thisBlock.trackCircuitFailure;

       powerFail=thisBlock.powerFailure;

       stationState=0;

       currAnnoun="";

       tperiod=1;

       ukprev=0;

       prerr=0;

       opstate=false;

       nextName=null;

       checkid=-1;

       power=0;

       overR=new ResponseUI(0, false, 0, 0, false, 0, false, 0, false, 0, false, 0, 0,

           0, 0, 0, 0, 0, 0, 0, false);

   }

   // Turns lights on and off

   public void setLights()

   {

       if((!thisBlock.isUground)&&(time<68400)&&(time>21600)

       &&(!oldBlock.isUground))

       {

           lights=false;

       }

       else

       {

           lights=true;

       }

   }

   // set safe power, set brakes if needed

   public PowJar setPower()

   {

       // coast if negative power request

       double thispower=0;

       double thisuk=0;

       double thiserr=0;

       boolean thiseBrake=eBrake;

       boolean thissBrake=sBrake;

       if(currspeed==0)

       {

           thisuk=0;

       }

       if(railFail||trackFail)

       {

           // cuts engine power, activates the eBrake, and releases passengers after a stop

           // has been reached for more serious problems.

           thiseBrake=true;

       }

       else if(powerFail)

       {

           // service brake activates if there is a power failure, train will

           // resume when power is restured

           thissBrake=true;

       }

       else if(engineFail||brakeFail||signalFail)

       {

           // cuts engine power if the above problems are detected

           // releases passengers when stopped

       }

       else

       {

           if((currspeed < safespeed)&&(!eBrake))

           {

               //disengage service brake and close doors if needed

               thissBrake=false;

               doors=false;

               // increase power

               thispower=(safespeed-currspeed)\*Kp+Ki\*(ukprev+tperiod\*(safespeed-currspeed+prerr)/2);

               thiserr=safespeed-currspeed;

               // engine can't deliver more power than max

               if(power>maxpower)

               {

                   thispower=maxpower;

               }

               // update Uk otherwise

               else if(power<0)

               {

                   thispower=0;

               }

               else

               {

                   thisuk=ukprev+tperiod\*(safespeed-currspeed+prerr)/2;

               }

           }

           else if(currspeed+1>safespeed)

           {

               // engage service brake if going 1m/s or more over safe speed

               thissBrake=true;

               thiserr=safespeed-currspeed;

               thisuk=ukprev+tperiod\*(safespeed-currspeed+prerr)/2;

           }

           else

           {

               // coast if going over safe speed but less than 1m/s over

               thissBrake=false;

               thiserr=safespeed-currspeed;

               thisuk=ukprev+tperiod\*(safespeed-currspeed+prerr)/2;

           }

       }

       PowJar currJar=new PowJar(thispower, thisuk, thiserr, thiseBrake, thissBrake);

       return currJar;

   }

   // sets safe speed

   public double setSafeSpeed()

   {

       double thisspeed=0;

       //checks which speed is the safest

       if((autspeed<slimit)&&(autspeed<setpoint))

       {

           thisspeed=autspeed;

       }

       else if((slimit<autspeed)&&(slimit<setpoint))

       {

           thisspeed=slimit;

       }

       else

       {

           thisspeed=setpoint;

       }

       //slows down if approaching station

       if((nextName!=null)&&(nextName.length()!=0)&&(thisspeed>5))

       {

           thisspeed=5;

       }

       return thisspeed;

   }

   // timeTick method that updates Train Controller from Train Model

   public ResponseTNC timeTick(double ntime, double curVelocity, double period, Block positionBlock,

   Block positionBlockTail, boolean issetSignalPickupFailure, boolean issetEngineFailure,

   boolean issetBrakeFailure, double fixedSuggestedSpeed, double mboSuggestedSpeed,

   boolean issetEmerBrake, boolean operator, String nextStationName)

   {

      // do nothing if paused

      if(!overR.paused)

      {

          //update info from Train Model

          time=ntime;

          tperiod=period;

          currspeed=curVelocity;

          thisBlock=positionBlock;

          oldBlock=positionBlockTail;

          signalFail=issetSignalPickupFailure;

          engineFail=issetEngineFailure;

          opstate=operator;

          brakeFail=issetBrakeFailure;

          railFail=positionBlock.brokenRailFailure;

          trackFail=positionBlock.trackCircuitFailure;

          powerFail=positionBlock.powerFailure;

          //autspeed=fixedSuggestedSpeed;

          if(mboSuggestedSpeed<autspeed)

          {

          //    autspeed=mboSuggestedSpeed;

          }

          slimit=positionBlock.speedLimit;

          nextName=nextStationName;

          currAnnoun=setAnnouncement();

          overRider();

          // begin redundant PLC actions for critical systems

          spone=setSafeSpeed();

          sptwo=setSafeSpeed();

          spthree=setSafeSpeed();

          safespeed=resolveSpeed(spone, sptwo, spthree);

          overRider();

          powone=setPower();

          powtwo=setPower();

          powthree=setPower();

          power=resolvePower(powone, powtwo, powthree);

          overRider();

          // do noncritical calculations

          setLights();

          checkStation();

          setDoors();

      }

      // send response to Train Model

      ResponseTNC tnmSignal=new ResponseTNC(power, sBrake, eBrake, lights, doors, tTemp, currAnnoun);

      return tnmSignal;

   }

   // takes values from UI

   public void overRider()

   {

       //updates values from operator

       setpoint=overR.currSetpoint;

       tTemp=overR.cit;

       //checks overrides and updates if they are engaged

       if(overR.cvoT)

       {

           currspeed=overR.cvo;

       }

       if(overR.sloT)

       {

           slimit=overR.slo;

       }

       if(overR.autOT)

       {

           autspeed=overR.autO;

       }

       if(overR.tbnOT)

       {

           if(overR.tbnO==-1)

           {

               thisBlock=negone;

           }

           else

           {

               thisBlock=negtwo;

           }

       }

       if(overR.timeOT)

       {

           time=overR.timeO\*3600;

       }

       if(overR.signalPFoT==1)

       {

           signalFail=true;

       }

       else if(overR.signalPFoT==2)

       {

           signalFail=false;

       }

       if(overR.brakeFoT==1)

       {

           brakeFail=true;

       }

       else if(overR.brakeFoT==2)

       {

           brakeFail=false;

       }

       if(overR.brokenRoT==1)

       {

           railFail=true;

       }

       else if(overR.brokenRoT==2)

       {

           railFail=false;

       }

       if(overR.trackCFoT==1)

       {

           trackFail=true;

       }

       else if(overR.trackCFoT==2)

       {

           trackFail=false;

       }

       if(overR.powerFoT==1)

       {

           powerFail=true;

       }

       else if(overR.powerFoT==2)

       {

           powerFail=false;

       }

       if(overR.sBrakeOT==1)

       {

           sBrake=true;

       }

       else if(overR.sBrakeOT==2)

       {

           sBrake=false;

       }

       if(overR.eBrakeOT==1)

       {

           eBrake=true;

       }

       else if(overR.eBrakeOT==2)

       {

           eBrake=false;

       }

       if(overR.trainEFoT==1)

       {

           engineFail=true;

       }

       else if(overR.trainEFoT==2)

       {

           engineFail=false;

       }

   }

   // stores UI values if selected

   public void updateUI(ResponseUI newR)

   {

       overR=newR;

   }

   // resolves any PLC conflicts about proper safe speed

   public double resolveSpeed(double firsts, double seconds, double thirds)

   {

       // ignores differences past 3 decimal places

       int firsti=(int)(Math.round(firsts\*1000));

       int secondi=(int)(Math.round(seconds\*1000));

       int thirdi=(int)(Math.round(thirds\*1000));

       //compares PLC values

       if((firsti==secondi)||(firsti==thirdi))

       {

           return firsts;

       }

       else if(secondi==thirdi)

       {

           return seconds;

       }

       else

       {

           // if two or more PLCs are broken, the safe speed is set to 0

           System.out.println("Error: No Majority PLC speed");

           return 0;

       }

   }

   // resolves any PLC conflicts about proper power output

   public double resolvePower(PowJar firsts, PowJar seconds, PowJar thirds)

   {

       // ignores differences past 3 decimal places

       int firsti=(int)(Math.round(firsts.jPower\*1000));

       int secondi=(int)(Math.round(seconds.jPower\*1000));

       int thirdi=(int)(Math.round(thirds.jPower\*1000));

       //compares PLC values

       if((firsti==secondi)||(firsti==thirdi))

       {

           eBrake=firsts.jEBrake;

           sBrake=firsts.jSBrake;

           ukprev=firsts.jUk;

           prerr=firsts.jErr;

           return firsts.jPower;

       }

       else if(secondi==thirdi)

       {

           eBrake=seconds.jEBrake;

           sBrake=seconds.jSBrake;

           ukprev=seconds.jUk;

           prerr=seconds.jErr;

           return seconds.jPower;

       }

       else

       {

           // if two or more PLCs are broken, the power is set to 0 and the emergency brake is engaged

           System.out.println("Error: No Majority PLC power");

           eBrake=true;

           return 0;

       }

   }

   // sets the doors open when the train is not moving

   public void setDoors()

   {

       if((currspeed!=0)||(stationState==2))

       {

           doors=false;

       }

       else

       {

           doors=true;

       }

   }

   // sets the current announcements

   public String setAnnouncement()

   {

       String thisAnnounce="Announcement Error";

       if((thisBlock.stationName==null)||(thisBlock.stationName.length()==0))

       {

           if((nextName==null)||(nextName.length()==0))

           {

               thisAnnounce="";

           }

           else

           {

               thisAnnounce="Next Station: "+nextName;

           }

       }

       else

       {

           thisAnnounce="Now arriving at "+thisBlock.stationName+" Station";

       }

       return thisAnnounce;

   }

   // takes action if approaching station

   public void checkStation()

   {

       if(thisBlock.isStation||thisBlock.isYard)

       {

           if(!opstate)

           {

               power=0;

               sBrake=true;

           }

           else if(stationState==0)

           {

               power=0;

               sBrake=true;

               if(currspeed==0)

               {

                   dtime=time;

                   stationState++;

               }

           }

       }

       if(stationState==1)

       {

           power=0;

           sBrake=true;

           if(time>=dtime+35)

           {

               doors=false;

               stationState++;

           }

       }

       else if(checkid!=thisBlock.id)

       {

           stationState=0;

           checkid=thisBlock.id;

       }

   }

}

**Class TNC\_UI:**

package TNM;

/\*\*

\* NSECS-TNC USER INTERFACE

\* USE EITHER AS PART OF WHOLE CONTROL SYSTEM

\* OR AS A TESTING PLATFORM BY RUNNING TNC\_TEST MODULE

\*

\* @author (Ben Tomasulo)

\* @version (4/24/2013)

\*/

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import TKM.\*;

public class TNC\_UI

{

JFrame theWindow; // This is the main window

private JTextField txtUser;

private JTextField txtUser\_1;

private JTextField txtUser\_2;

private JTextField txtUser\_3;

private JTextField txtUser\_4;

private JTextField txtUser\_5;

private JTextField txtUser\_7;

private JTextField txtUser\_8;

//initialize field variables

private int userTrain;

public int currTrain;

private double currVel;

private double cvo;

private boolean cvoT;

private double currPow;

private double currSetpoint;

private double userSetpoint;

private double speedLimit;

private double slo;

private boolean sloT;

private double aut;

private double autO;

private boolean autOT;

private double plcVel;

private int tbn;

private int tbnO;

private boolean tbnOT;

private double cit;

private double userIdealT;

private int timeH;

private int timeM;

private int timeS;

private double timeO;

private boolean timeOT;

private boolean pause;

private boolean tick;

private boolean trainEF;

private int trainEFoT;

private boolean signalPF;

private int signalPFoT;

private boolean brakeF;

private int brakeFoT;

private boolean brokenR;

private int brokenRoT;

private boolean trackCF;

private int trackCFoT;

private boolean powerF;

private int powerFoT;

private boolean sBrake;

private int sBrakeOT;

private boolean eBrake;

private int eBrakeOT;

private boolean doors;

private String announ;

private boolean lights;

public int lsize;

//initialize JLabels

private JLabel lblVel;

private JLabel label;

private JLabel lblPause;

private JLabel lblOnoff;

private JLabel lblEngaged\_CVOT;

private JLabel lblFail;

private JLabel lblPow;

private JLabel lblFail\_1;

private JLabel lblSetVel;

private JLabel lblFail\_2;

private JLabel lblFail\_3;

private JLabel lblVelLimit;

private JLabel lblFail\_4;

private JLabel lblFail\_5;

private JLabel lblEngaged\_5;

private JLabel lblFail\_6;

private JLabel lblAutVel;

private JLabel lblFail\_7;

private JLabel lblFail\_8;

private JLabel lblEngaged\_6;

private JLabel lblFail\_9;

private JLabel lblPlcVel;

private JLabel lblFail\_10;

private JLabel lblBlock;

private JLabel lblFail\_11;

private JLabel lblEngaged;

private JLabel lblBlockOverride;

private JLabel lblEngaged\_1;

private JLabel lblIdealtemp;

private JLabel lblClosed;

private JLabel lblTheRainIn;

private JLabel lblTime;

private JLabel lblLightsOn;

private JLabel lblEngaged\_2;

private JLabel lblTimeOverride;

private JLabel lblEngaged\_3;

//initialize JButtons

private JButton btnPauseresumeToggle;

private JButton btnSet\_1;

public TNC\_UI(boolean control)

{

// default field values

currTrain=1;

currVel=-1;

cvoT=false;

currPow=-1;

currSetpoint=0;

speedLimit=-1;

sloT=false;

aut=-1;

autO=-1;

autOT=false;

plcVel=-1;

tbn=-1;

tbnO=-1;

tbnOT=false;

cit=22;

timeH=-1;

timeM=-1;

timeS=-1;

timeO=0;

timeOT=false;

pause=false;

tick=control;

trainEF=false;

trainEFoT=0;

signalPF=false;

signalPFoT=0;

brakeF=false;

brakeFoT=0;

trackCF=false;

trackCFoT=0;

powerF=false;

powerFoT=0;

sBrake=false;

sBrakeOT=0;

eBrake=false;

eBrakeOT=0;

doors=false;

announ="-1";

lights=false;

lsize=1;

// create display window

theWindow = new JFrame("NSECS-TNC USER INTERFACE");

theWindow.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

theWindow.setSize(950,750);

GridBagLayout gridBagLayout = new GridBagLayout();

gridBagLayout.columnWidths = new int[]{278, 134, 75, 32, 278, 108, 0};

gridBagLayout.rowHeights = new int[] {30, 29, 30, 29, 29, 16, 30, 16, 30, 29, 29, 30, 29,

16, 29, 30, 29, 16, 30, 29, 16, 30};

gridBagLayout.columnWeights = new double[]{0.0, 0.0, 0.0, 0.0, 0.0, 0.0, Double.MIN\_VALUE};

gridBagLayout.rowWeights = new double[]{0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0};

theWindow.getContentPane().setLayout(gridBagLayout);

// display window information

JLabel lblSelectTrain = new JLabel("Select Train:");

GridBagConstraints gbc\_lblSelectTrain = new GridBagConstraints();

gbc\_lblSelectTrain.anchor = GridBagConstraints.EAST;

gbc\_lblSelectTrain.insets = new Insets(0, 0, 5, 5);

gbc\_lblSelectTrain.gridx = 0;

gbc\_lblSelectTrain.gridy = 0;

theWindow.getContentPane().add(lblSelectTrain, gbc\_lblSelectTrain);

txtUser = new JTextField();

txtUser.setText("user #");

GridBagConstraints gbc\_txtUser = new GridBagConstraints();

gbc\_txtUser.anchor = GridBagConstraints.NORTH;

gbc\_txtUser.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser.gridx = 1;

gbc\_txtUser.gridy = 0;

theWindow.getContentPane().add(txtUser, gbc\_txtUser);

txtUser.setColumns(10);

JButton btnSet = new JButton("set");

GridBagConstraints gbc\_btnSet = new GridBagConstraints();

gbc\_btnSet.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet.gridx = 2;

gbc\_btnSet.gridy = 0;

theWindow.getContentPane().add(btnSet, gbc\_btnSet);

btnSet.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

int currInt=Integer.parseInt(txtUser.getText());

if((currInt>=0)&&(currInt<=lsize))

{

currTrain=currInt;

}

}

});

JLabel lblCurrentTrainSelected = new JLabel("Current Train Selected:");

GridBagConstraints gbc\_lblCurrentTrainSelected = new GridBagConstraints();

gbc\_lblCurrentTrainSelected.anchor = GridBagConstraints.EAST;

gbc\_lblCurrentTrainSelected.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentTrainSelected.gridx = 4;

gbc\_lblCurrentTrainSelected.gridy = 0;

theWindow.getContentPane().add(lblCurrentTrainSelected, gbc\_lblCurrentTrainSelected);

label = new JLabel(""+currTrain);

GridBagConstraints gbc\_label = new GridBagConstraints();

gbc\_label.anchor = GridBagConstraints.WEST;

gbc\_label.insets = new Insets(0, 0, 5, 0);

gbc\_label.gridx = 5;

gbc\_label.gridy = 0;

theWindow.getContentPane().add(label, gbc\_label);

JLabel lblCurrentVelocitykmh = new JLabel("Current Velocity (m/s):");

GridBagConstraints gbc\_lblCurrentVelocitykmh = new GridBagConstraints();

gbc\_lblCurrentVelocitykmh.anchor = GridBagConstraints.EAST;

gbc\_lblCurrentVelocitykmh.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentVelocitykmh.gridx = 0;

gbc\_lblCurrentVelocitykmh.gridy = 1;

theWindow.getContentPane().add(lblCurrentVelocitykmh, gbc\_lblCurrentVelocitykmh);

lblVel = new JLabel(""+currVel);

GridBagConstraints gbc\_lblVel = new GridBagConstraints();

gbc\_lblVel.anchor = GridBagConstraints.WEST;

gbc\_lblVel.insets = new Insets(0, 0, 5, 5);

gbc\_lblVel.gridx = 1;

gbc\_lblVel.gridy = 1;

theWindow.getContentPane().add(lblVel, gbc\_lblVel);

btnPauseresumeToggle = new JButton("Pause/Resume Toggle");

GridBagConstraints gbc\_btnPauseresumeToggle = new GridBagConstraints();

gbc\_btnPauseresumeToggle.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnPauseresumeToggle.insets = new Insets(0, 0, 5, 5);

gbc\_btnPauseresumeToggle.gridx = 4;

gbc\_btnPauseresumeToggle.gridy = 1;

theWindow.getContentPane().add(btnPauseresumeToggle, gbc\_btnPauseresumeToggle);

btnPauseresumeToggle.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

pause=bToggle(pause);

}

});

lblPause = new JLabel(bPause(pause));

GridBagConstraints gbc\_lblPause = new GridBagConstraints();

gbc\_lblPause.anchor = GridBagConstraints.WEST;

gbc\_lblPause.insets = new Insets(0, 0, 5, 0);

gbc\_lblPause.gridx = 5;

gbc\_lblPause.gridy = 1;

theWindow.getContentPane().add(lblPause, gbc\_lblPause);

JLabel lblCurrentVelocityOverride = new JLabel("Current Velocity Override (m/s):");

GridBagConstraints gbc\_lblCurrentVelocityOverride = new GridBagConstraints();

gbc\_lblCurrentVelocityOverride.anchor = GridBagConstraints.EAST;

gbc\_lblCurrentVelocityOverride.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentVelocityOverride.gridx = 0;

gbc\_lblCurrentVelocityOverride.gridy = 2;

theWindow.getContentPane().add(lblCurrentVelocityOverride, gbc\_lblCurrentVelocityOverride);

txtUser\_1 = new JTextField();

txtUser\_1.setText("user #");

GridBagConstraints gbc\_txtUser\_1 = new GridBagConstraints();

gbc\_txtUser\_1.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_1.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_1.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_1.gridx = 1;

gbc\_txtUser\_1.gridy = 2;

theWindow.getContentPane().add(txtUser\_1, gbc\_txtUser\_1);

txtUser\_1.setColumns(10);

btnSet\_1 = new JButton("set");

GridBagConstraints gbc\_btnSet\_1 = new GridBagConstraints();

gbc\_btnSet\_1.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_1.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_1.gridx = 2;

gbc\_btnSet\_1.gridy = 2;

theWindow.getContentPane().add(btnSet\_1, gbc\_btnSet\_1);

btnSet\_1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

double currDou=Double.parseDouble(txtUser\_1.getText());

cvo=currDou;

}

});

JLabel lblTickOverride = new JLabel("Tick Override:");

GridBagConstraints gbc\_lblTickOverride = new GridBagConstraints();

gbc\_lblTickOverride.anchor = GridBagConstraints.EAST;

gbc\_lblTickOverride.insets = new Insets(0, 0, 5, 5);

gbc\_lblTickOverride.gridx = 4;

gbc\_lblTickOverride.gridy = 2;

theWindow.getContentPane().add(lblTickOverride, gbc\_lblTickOverride);

lblOnoff = new JLabel(bTick(tick));

GridBagConstraints gbc\_lblOnoff = new GridBagConstraints();

gbc\_lblOnoff.anchor = GridBagConstraints.WEST;

gbc\_lblOnoff.insets = new Insets(0, 0, 5, 0);

gbc\_lblOnoff.gridx = 5;

gbc\_lblOnoff.gridy = 2;

theWindow.getContentPane().add(lblOnoff, gbc\_lblOnoff);

JButton btnCurrentVelocityOverride = new JButton("Current Velocity Override Toggle");

GridBagConstraints gbc\_btnCurrentVelocityOverride = new GridBagConstraints();

gbc\_btnCurrentVelocityOverride.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnCurrentVelocityOverride.insets = new Insets(0, 0, 5, 5);

gbc\_btnCurrentVelocityOverride.gridx = 0;

gbc\_btnCurrentVelocityOverride.gridy = 3;

theWindow.getContentPane().add(btnCurrentVelocityOverride, gbc\_btnCurrentVelocityOverride);

btnCurrentVelocityOverride.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

cvoT=bToggle(cvoT);

}

});

lblEngaged\_CVOT = new JLabel(bTout(cvoT));

GridBagConstraints gbc\_lblEngaged\_CVOT = new GridBagConstraints();

gbc\_lblEngaged\_CVOT.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged\_CVOT.insets = new Insets(0, 0, 5, 5);

gbc\_lblEngaged\_CVOT.gridx = 1;

gbc\_lblEngaged\_CVOT.gridy = 3;

theWindow.getContentPane().add(lblEngaged\_CVOT, gbc\_lblEngaged\_CVOT);

JLabel lblTrainEngineFailure = new JLabel("Train Engine Failure:");

GridBagConstraints gbc\_lblTrainEngineFailure = new GridBagConstraints();

gbc\_lblTrainEngineFailure.anchor = GridBagConstraints.EAST;

gbc\_lblTrainEngineFailure.insets = new Insets(0, 0, 5, 5);

gbc\_lblTrainEngineFailure.gridx = 4;

gbc\_lblTrainEngineFailure.gridy = 3;

theWindow.getContentPane().add(lblTrainEngineFailure, gbc\_lblTrainEngineFailure);

lblFail = new JLabel(bFout(trainEF));

GridBagConstraints gbc\_lblFail = new GridBagConstraints();

gbc\_lblFail.anchor = GridBagConstraints.WEST;

gbc\_lblFail.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail.gridx = 5;

gbc\_lblFail.gridy = 3;

theWindow.getContentPane().add(lblFail, gbc\_lblFail);

JLabel lblCurrentPowerw = new JLabel("Current Power (W):");

GridBagConstraints gbc\_lblCurrentPowerw = new GridBagConstraints();

gbc\_lblCurrentPowerw.anchor = GridBagConstraints.EAST;

gbc\_lblCurrentPowerw.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentPowerw.gridx = 0;

gbc\_lblCurrentPowerw.gridy = 4;

theWindow.getContentPane().add(lblCurrentPowerw, gbc\_lblCurrentPowerw);

lblPow = new JLabel(""+currPow);

GridBagConstraints gbc\_lblPow = new GridBagConstraints();

gbc\_lblPow.anchor = GridBagConstraints.WEST;

gbc\_lblPow.insets = new Insets(0, 0, 5, 5);

gbc\_lblPow.gridx = 1;

gbc\_lblPow.gridy = 4;

theWindow.getContentPane().add(lblPow, gbc\_lblPow);

JButton btnTrainEngineFailiure = new JButton("Train Engine Failure Override Toggle");

GridBagConstraints gbc\_btnTrainEngineFailiure = new GridBagConstraints();

gbc\_btnTrainEngineFailiure.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnTrainEngineFailiure.insets = new Insets(0, 0, 5, 5);

gbc\_btnTrainEngineFailiure.gridx = 4;

gbc\_btnTrainEngineFailiure.gridy = 4;

theWindow.getContentPane().add(btnTrainEngineFailiure, gbc\_btnTrainEngineFailiure);

btnTrainEngineFailiure.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

trainEFoT=iToggle(trainEFoT);

}

});

lblFail\_1 = new JLabel(iTfail(trainEFoT));

GridBagConstraints gbc\_lblFail\_1 = new GridBagConstraints();

gbc\_lblFail\_1.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_1.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_1.gridx = 5;

gbc\_lblFail\_1.gridy = 4;

theWindow.getContentPane().add(lblFail\_1, gbc\_lblFail\_1);

JLabel lblCurrentSetpointVelocity = new JLabel("Current Setpoint Velocity (m/s):");

GridBagConstraints gbc\_lblCurrentSetpointVelocity = new GridBagConstraints();

gbc\_lblCurrentSetpointVelocity.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblCurrentSetpointVelocity.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentSetpointVelocity.gridx = 0;

gbc\_lblCurrentSetpointVelocity.gridy = 5;

theWindow.getContentPane().add(lblCurrentSetpointVelocity, gbc\_lblCurrentSetpointVelocity);

lblSetVel = new JLabel(""+currSetpoint);

GridBagConstraints gbc\_lblSetVel = new GridBagConstraints();

gbc\_lblSetVel.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblSetVel.insets = new Insets(0, 0, 5, 5);

gbc\_lblSetVel.gridx = 1;

gbc\_lblSetVel.gridy = 5;

theWindow.getContentPane().add(lblSetVel, gbc\_lblSetVel);

JLabel lblSignalPickupFailure = new JLabel("Signal Pickup Failure:");

GridBagConstraints gbc\_lblSignalPickupFailure = new GridBagConstraints();

gbc\_lblSignalPickupFailure.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblSignalPickupFailure.insets = new Insets(0, 0, 5, 5);

gbc\_lblSignalPickupFailure.gridx = 4;

gbc\_lblSignalPickupFailure.gridy = 5;

theWindow.getContentPane().add(lblSignalPickupFailure, gbc\_lblSignalPickupFailure);

lblFail\_2 = new JLabel(bFout(signalPF));

GridBagConstraints gbc\_lblFail\_2 = new GridBagConstraints();

gbc\_lblFail\_2.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblFail\_2.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_2.gridx = 5;

gbc\_lblFail\_2.gridy = 5;

theWindow.getContentPane().add(lblFail\_2, gbc\_lblFail\_2);

JLabel lblSetpointVelocitykmh = new JLabel("Setpoint Velocity (m/s):");

GridBagConstraints gbc\_lblSetpointVelocitykmh = new GridBagConstraints();

gbc\_lblSetpointVelocitykmh.anchor = GridBagConstraints.EAST;

gbc\_lblSetpointVelocitykmh.insets = new Insets(0, 0, 5, 5);

gbc\_lblSetpointVelocitykmh.gridx = 0;

gbc\_lblSetpointVelocitykmh.gridy = 6;

theWindow.getContentPane().add(lblSetpointVelocitykmh, gbc\_lblSetpointVelocitykmh);

txtUser\_2 = new JTextField();

txtUser\_2.setText("user #");

GridBagConstraints gbc\_txtUser\_2 = new GridBagConstraints();

gbc\_txtUser\_2.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_2.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_2.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_2.gridx = 1;

gbc\_txtUser\_2.gridy = 6;

theWindow.getContentPane().add(txtUser\_2, gbc\_txtUser\_2);

txtUser\_2.setColumns(10);

JButton btnSet\_2 = new JButton("set");

GridBagConstraints gbc\_btnSet\_2 = new GridBagConstraints();

gbc\_btnSet\_2.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_2.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_2.gridx = 2;

gbc\_btnSet\_2.gridy = 6;

theWindow.getContentPane().add(btnSet\_2, gbc\_btnSet\_2);

btnSet\_2.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

double currDou=Double.parseDouble(txtUser\_2.getText());

currSetpoint=currDou;

}

});

JButton btnSignalPickupFailure = new JButton("Signal Pickup Failure Override Toggle");

GridBagConstraints gbc\_btnSignalPickupFailure = new GridBagConstraints();

gbc\_btnSignalPickupFailure.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSignalPickupFailure.insets = new Insets(0, 0, 5, 5);

gbc\_btnSignalPickupFailure.gridx = 4;

gbc\_btnSignalPickupFailure.gridy = 6;

theWindow.getContentPane().add(btnSignalPickupFailure, gbc\_btnSignalPickupFailure);

btnSignalPickupFailure.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

signalPFoT=iToggle(signalPFoT);

}

});

lblFail\_3 = new JLabel(iTfail(signalPFoT));

GridBagConstraints gbc\_lblFail\_3 = new GridBagConstraints();

gbc\_lblFail\_3.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_3.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_3.gridx = 5;

gbc\_lblFail\_3.gridy = 6;

theWindow.getContentPane().add(lblFail\_3, gbc\_lblFail\_3);

JLabel lblSpeedLimitkmh = new JLabel("Speed Limit (m/s):");

GridBagConstraints gbc\_lblSpeedLimitkmh = new GridBagConstraints();

gbc\_lblSpeedLimitkmh.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblSpeedLimitkmh.insets = new Insets(0, 0, 5, 5);

gbc\_lblSpeedLimitkmh.gridx = 0;

gbc\_lblSpeedLimitkmh.gridy = 7;

theWindow.getContentPane().add(lblSpeedLimitkmh, gbc\_lblSpeedLimitkmh);

lblVelLimit = new JLabel(""+speedLimit);

GridBagConstraints gbc\_lblVelLimit = new GridBagConstraints();

gbc\_lblVelLimit.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblVelLimit.insets = new Insets(0, 0, 5, 5);

gbc\_lblVelLimit.gridx = 1;

gbc\_lblVelLimit.gridy = 7;

theWindow.getContentPane().add(lblVelLimit, gbc\_lblVelLimit);

JLabel lblBrakeFailure = new JLabel("Brake Failure:");

GridBagConstraints gbc\_lblBrakeFailure = new GridBagConstraints();

gbc\_lblBrakeFailure.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblBrakeFailure.insets = new Insets(0, 0, 5, 5);

gbc\_lblBrakeFailure.gridx = 4;

gbc\_lblBrakeFailure.gridy = 7;

theWindow.getContentPane().add(lblBrakeFailure, gbc\_lblBrakeFailure);

lblFail\_4 = new JLabel(bFout(brakeF));

GridBagConstraints gbc\_lblFail\_4 = new GridBagConstraints();

gbc\_lblFail\_4.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblFail\_4.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_4.gridx = 5;

gbc\_lblFail\_4.gridy = 7;

theWindow.getContentPane().add(lblFail\_4, gbc\_lblFail\_4);

JLabel lblSpeedLimitOverride = new JLabel("Speed Limit Override (m/s):");

GridBagConstraints gbc\_lblSpeedLimitOverride = new GridBagConstraints();

gbc\_lblSpeedLimitOverride.anchor = GridBagConstraints.EAST;

gbc\_lblSpeedLimitOverride.insets = new Insets(0, 0, 5, 5);

gbc\_lblSpeedLimitOverride.gridx = 0;

gbc\_lblSpeedLimitOverride.gridy = 8;

theWindow.getContentPane().add(lblSpeedLimitOverride, gbc\_lblSpeedLimitOverride);

txtUser\_3 = new JTextField();

txtUser\_3.setText("user #");

GridBagConstraints gbc\_txtUser\_3 = new GridBagConstraints();

gbc\_txtUser\_3.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_3.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_3.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_3.gridx = 1;

gbc\_txtUser\_3.gridy = 8;

theWindow.getContentPane().add(txtUser\_3, gbc\_txtUser\_3);

txtUser\_3.setColumns(10);

JButton btnSet\_3 = new JButton("set");

GridBagConstraints gbc\_btnSet\_3 = new GridBagConstraints();

gbc\_btnSet\_3.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_3.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_3.gridx = 2;

gbc\_btnSet\_3.gridy = 8;

theWindow.getContentPane().add(btnSet\_3, gbc\_btnSet\_3);

btnSet\_3.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

double currDou=Double.parseDouble(txtUser\_3.getText());

slo=currDou;

}

});

JButton btnBrakeFailureOverride = new JButton("Brake Failure Override Toggle");

GridBagConstraints gbc\_btnBrakeFailureOverride = new GridBagConstraints();

gbc\_btnBrakeFailureOverride.anchor = GridBagConstraints.SOUTHEAST;

gbc\_btnBrakeFailureOverride.insets = new Insets(0, 0, 5, 5);

gbc\_btnBrakeFailureOverride.gridx = 4;

gbc\_btnBrakeFailureOverride.gridy = 8;

theWindow.getContentPane().add(btnBrakeFailureOverride, gbc\_btnBrakeFailureOverride);

btnBrakeFailureOverride.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

brakeFoT=iToggle(brakeFoT);

}

});

lblFail\_5 = new JLabel(iTfail(brakeFoT));

GridBagConstraints gbc\_lblFail\_5 = new GridBagConstraints();

gbc\_lblFail\_5.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_5.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_5.gridx = 5;

gbc\_lblFail\_5.gridy = 8;

theWindow.getContentPane().add(lblFail\_5, gbc\_lblFail\_5);

JButton btnSpeedLimitOverride = new JButton("Speed Limit Override Toggle");

GridBagConstraints gbc\_btnSpeedLimitOverride = new GridBagConstraints();

gbc\_btnSpeedLimitOverride.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnSpeedLimitOverride.insets = new Insets(0, 0, 5, 5);

gbc\_btnSpeedLimitOverride.gridx = 0;

gbc\_btnSpeedLimitOverride.gridy = 9;

theWindow.getContentPane().add(btnSpeedLimitOverride, gbc\_btnSpeedLimitOverride);

btnSpeedLimitOverride.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

sloT=bToggle(sloT);

}

});

lblEngaged\_5 = new JLabel(bTout(sloT));

GridBagConstraints gbc\_lblEngaged\_5 = new GridBagConstraints();

gbc\_lblEngaged\_5.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged\_5.insets = new Insets(0, 0, 5, 5);

gbc\_lblEngaged\_5.gridx = 1;

gbc\_lblEngaged\_5.gridy = 9;

theWindow.getContentPane().add(lblEngaged\_5, gbc\_lblEngaged\_5);

JLabel lblBrokenRail = new JLabel("Broken Rail:");

GridBagConstraints gbc\_lblBrokenRail = new GridBagConstraints();

gbc\_lblBrokenRail.anchor = GridBagConstraints.EAST;

gbc\_lblBrokenRail.insets = new Insets(0, 0, 5, 5);

gbc\_lblBrokenRail.gridx = 4;

gbc\_lblBrokenRail.gridy = 9;

theWindow.getContentPane().add(lblBrokenRail, gbc\_lblBrokenRail);

lblFail\_6 = new JLabel(bFout(brokenR));

GridBagConstraints gbc\_lblFail\_6 = new GridBagConstraints();

gbc\_lblFail\_6.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_6.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_6.gridx = 5;

gbc\_lblFail\_6.gridy = 9;

theWindow.getContentPane().add(lblFail\_6, gbc\_lblFail\_6);

JLabel lblAuthoritykmh = new JLabel("Authority (m/s):");

GridBagConstraints gbc\_lblAuthoritykmh = new GridBagConstraints();

gbc\_lblAuthoritykmh.anchor = GridBagConstraints.EAST;

gbc\_lblAuthoritykmh.insets = new Insets(0, 0, 5, 5);

gbc\_lblAuthoritykmh.gridx = 0;

gbc\_lblAuthoritykmh.gridy = 10;

theWindow.getContentPane().add(lblAuthoritykmh, gbc\_lblAuthoritykmh);

lblAutVel = new JLabel(""+aut);

GridBagConstraints gbc\_lblAutVel = new GridBagConstraints();

gbc\_lblAutVel.anchor = GridBagConstraints.WEST;

gbc\_lblAutVel.insets = new Insets(0, 0, 5, 5);

gbc\_lblAutVel.gridx = 1;

gbc\_lblAutVel.gridy = 10;

theWindow.getContentPane().add(lblAutVel, gbc\_lblAutVel);

JButton btnBrokenRailOverride = new JButton("Broken Rail Override Toggle");

GridBagConstraints gbc\_btnBrokenRailOverride = new GridBagConstraints();

gbc\_btnBrokenRailOverride.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnBrokenRailOverride.insets = new Insets(0, 0, 5, 5);

gbc\_btnBrokenRailOverride.gridx = 4;

gbc\_btnBrokenRailOverride.gridy = 10;

theWindow.getContentPane().add(btnBrokenRailOverride, gbc\_btnBrokenRailOverride);

btnBrokenRailOverride.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

brokenRoT=iToggle(brokenRoT);

}

});

lblFail\_7 = new JLabel(iTfail(brokenRoT));

GridBagConstraints gbc\_lblFail\_7 = new GridBagConstraints();

gbc\_lblFail\_7.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_7.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_7.gridx = 5;

gbc\_lblFail\_7.gridy = 10;

theWindow.getContentPane().add(lblFail\_7, gbc\_lblFail\_7);

JLabel lblAuthorityOverridekmh = new JLabel("Authority Override (m/s):");

GridBagConstraints gbc\_lblAuthorityOverridekmh = new GridBagConstraints();

gbc\_lblAuthorityOverridekmh.anchor = GridBagConstraints.EAST;

gbc\_lblAuthorityOverridekmh.insets = new Insets(0, 0, 5, 5);

gbc\_lblAuthorityOverridekmh.gridx = 0;

gbc\_lblAuthorityOverridekmh.gridy = 11;

theWindow.getContentPane().add(lblAuthorityOverridekmh, gbc\_lblAuthorityOverridekmh);

txtUser\_4 = new JTextField();

txtUser\_4.setText("user #");

GridBagConstraints gbc\_txtUser\_4 = new GridBagConstraints();

gbc\_txtUser\_4.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_4.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_4.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_4.gridx = 1;

gbc\_txtUser\_4.gridy = 11;

theWindow.getContentPane().add(txtUser\_4, gbc\_txtUser\_4);

txtUser\_4.setColumns(10);

JButton btnSet\_4 = new JButton("set");

GridBagConstraints gbc\_btnSet\_4 = new GridBagConstraints();

gbc\_btnSet\_4.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_4.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_4.gridx = 2;

gbc\_btnSet\_4.gridy = 11;

theWindow.getContentPane().add(btnSet\_4, gbc\_btnSet\_4);

btnSet\_4.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

double currDou=Double.parseDouble(txtUser\_4.getText());

autO=currDou;

}

});

JLabel lblTrackCircuitFailure = new JLabel("Track Circuit Failure:");

GridBagConstraints gbc\_lblTrackCircuitFailure = new GridBagConstraints();

gbc\_lblTrackCircuitFailure.anchor = GridBagConstraints.EAST;

gbc\_lblTrackCircuitFailure.insets = new Insets(0, 0, 5, 5);

gbc\_lblTrackCircuitFailure.gridx = 4;

gbc\_lblTrackCircuitFailure.gridy = 11;

theWindow.getContentPane().add(lblTrackCircuitFailure, gbc\_lblTrackCircuitFailure);

lblFail\_8 = new JLabel(bFout(trackCF));

GridBagConstraints gbc\_lblFail\_8 = new GridBagConstraints();

gbc\_lblFail\_8.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_8.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_8.gridx = 5;

gbc\_lblFail\_8.gridy = 11;

theWindow.getContentPane().add(lblFail\_8, gbc\_lblFail\_8);

JButton btnAuthorityOverrideToggle = new JButton("Authority Override Toggle");

GridBagConstraints gbc\_btnAuthorityOverrideToggle = new GridBagConstraints();

gbc\_btnAuthorityOverrideToggle.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnAuthorityOverrideToggle.insets = new Insets(0, 0, 5, 5);

gbc\_btnAuthorityOverrideToggle.gridx = 0;

gbc\_btnAuthorityOverrideToggle.gridy = 12;

theWindow.getContentPane().add(btnAuthorityOverrideToggle, gbc\_btnAuthorityOverrideToggle);

btnAuthorityOverrideToggle.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

autOT=bToggle(autOT);

}

});

lblEngaged\_6 = new JLabel(bTout(autOT));

GridBagConstraints gbc\_lblEngaged\_6 = new GridBagConstraints();

gbc\_lblEngaged\_6.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged\_6.insets = new Insets(0, 0, 5, 5);

gbc\_lblEngaged\_6.gridx = 1;

gbc\_lblEngaged\_6.gridy = 12;

theWindow.getContentPane().add(lblEngaged\_6, gbc\_lblEngaged\_6);

JButton btnTrackCircuitFailure = new JButton("Track Circuit Failure Override Toggle");

GridBagConstraints gbc\_btnTrackCircuitFailure = new GridBagConstraints();

gbc\_btnTrackCircuitFailure.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnTrackCircuitFailure.insets = new Insets(0, 0, 5, 5);

gbc\_btnTrackCircuitFailure.gridx = 4;

gbc\_btnTrackCircuitFailure.gridy = 12;

theWindow.getContentPane().add(btnTrackCircuitFailure, gbc\_btnTrackCircuitFailure);

btnTrackCircuitFailure.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

trackCFoT=iToggle(trackCFoT);

}

});

lblFail\_9 = new JLabel(iTfail(trackCFoT));

GridBagConstraints gbc\_lblFail\_9 = new GridBagConstraints();

gbc\_lblFail\_9.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_9.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_9.gridx = 5;

gbc\_lblFail\_9.gridy = 12;

theWindow.getContentPane().add(lblFail\_9, gbc\_lblFail\_9);

JLabel lblPlcChosenVelocity = new JLabel("PLC Chosen Velocity (m/s):");

GridBagConstraints gbc\_lblPlcChosenVelocity = new GridBagConstraints();

gbc\_lblPlcChosenVelocity.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblPlcChosenVelocity.insets = new Insets(0, 0, 5, 5);

gbc\_lblPlcChosenVelocity.gridx = 0;

gbc\_lblPlcChosenVelocity.gridy = 13;

theWindow.getContentPane().add(lblPlcChosenVelocity, gbc\_lblPlcChosenVelocity);

lblPlcVel = new JLabel(""+plcVel);

GridBagConstraints gbc\_lblPlcVel = new GridBagConstraints();

gbc\_lblPlcVel.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblPlcVel.insets = new Insets(0, 0, 5, 5);

gbc\_lblPlcVel.gridx = 1;

gbc\_lblPlcVel.gridy = 13;

theWindow.getContentPane().add(lblPlcVel, gbc\_lblPlcVel);

JLabel lblPowerFailure = new JLabel("Power Failure:");

GridBagConstraints gbc\_lblPowerFailure = new GridBagConstraints();

gbc\_lblPowerFailure.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblPowerFailure.insets = new Insets(0, 0, 5, 5);

gbc\_lblPowerFailure.gridx = 4;

gbc\_lblPowerFailure.gridy = 13;

theWindow.getContentPane().add(lblPowerFailure, gbc\_lblPowerFailure);

lblFail\_10 = new JLabel(bFout(powerF));

GridBagConstraints gbc\_lblFail\_10 = new GridBagConstraints();

gbc\_lblFail\_10.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblFail\_10.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_10.gridx = 5;

gbc\_lblFail\_10.gridy = 13;

theWindow.getContentPane().add(lblFail\_10, gbc\_lblFail\_10);

JLabel lblTrackBlockNumber = new JLabel("Track Block Number:");

GridBagConstraints gbc\_lblTrackBlockNumber = new GridBagConstraints();

gbc\_lblTrackBlockNumber.anchor = GridBagConstraints.EAST;

gbc\_lblTrackBlockNumber.insets = new Insets(0, 0, 5, 5);

gbc\_lblTrackBlockNumber.gridx = 0;

gbc\_lblTrackBlockNumber.gridy = 14;

theWindow.getContentPane().add(lblTrackBlockNumber, gbc\_lblTrackBlockNumber);

lblBlock = new JLabel(""+tbn);

GridBagConstraints gbc\_lblBlock = new GridBagConstraints();

gbc\_lblBlock.anchor = GridBagConstraints.WEST;

gbc\_lblBlock.insets = new Insets(0, 0, 5, 5);

gbc\_lblBlock.gridx = 1;

gbc\_lblBlock.gridy = 14;

theWindow.getContentPane().add(lblBlock, gbc\_lblBlock);

JButton btnPowerFailureOverride = new JButton("Power Failure Override Toggle");

GridBagConstraints gbc\_btnPowerFailureOverride = new GridBagConstraints();

gbc\_btnPowerFailureOverride.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnPowerFailureOverride.insets = new Insets(0, 0, 5, 5);

gbc\_btnPowerFailureOverride.gridx = 4;

gbc\_btnPowerFailureOverride.gridy = 14;

theWindow.getContentPane().add(btnPowerFailureOverride, gbc\_btnPowerFailureOverride);

btnPowerFailureOverride.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

powerFoT=iToggle(powerFoT);

}

});

lblFail\_11 = new JLabel(iTfail(powerFoT));

GridBagConstraints gbc\_lblFail\_11 = new GridBagConstraints();

gbc\_lblFail\_11.anchor = GridBagConstraints.WEST;

gbc\_lblFail\_11.insets = new Insets(0, 0, 5, 0);

gbc\_lblFail\_11.gridx = 5;

gbc\_lblFail\_11.gridy = 14;

theWindow.getContentPane().add(lblFail\_11, gbc\_lblFail\_11);

JLabel lblTrackBlockNumber\_1 = new JLabel("Track Block Number Override:");

GridBagConstraints gbc\_lblTrackBlockNumber\_1 = new GridBagConstraints();

gbc\_lblTrackBlockNumber\_1.anchor = GridBagConstraints.EAST;

gbc\_lblTrackBlockNumber\_1.insets = new Insets(0, 0, 5, 5);

gbc\_lblTrackBlockNumber\_1.gridx = 0;

gbc\_lblTrackBlockNumber\_1.gridy = 15;

theWindow.getContentPane().add(lblTrackBlockNumber\_1, gbc\_lblTrackBlockNumber\_1);

txtUser\_5 = new JTextField();

txtUser\_5.setText("user #");

GridBagConstraints gbc\_txtUser\_5 = new GridBagConstraints();

gbc\_txtUser\_5.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_5.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_5.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_5.gridx = 1;

gbc\_txtUser\_5.gridy = 15;

theWindow.getContentPane().add(txtUser\_5, gbc\_txtUser\_5);

txtUser\_5.setColumns(10);

JButton btnSet\_5 = new JButton("set");

GridBagConstraints gbc\_btnSet\_5 = new GridBagConstraints();

gbc\_btnSet\_5.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_5.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_5.gridx = 2;

gbc\_btnSet\_5.gridy = 15;

theWindow.getContentPane().add(btnSet\_5, gbc\_btnSet\_5);

btnSet\_5.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

int currInt=Integer.parseInt(txtUser\_5.getText());

if((currInt==-1)||(currInt==-2))

{

tbnO=currInt;

}

}

});

JLabel lblServiceBrake = new JLabel("Service Brake State:");

GridBagConstraints gbc\_lblServiceBrake = new GridBagConstraints();

gbc\_lblServiceBrake.anchor = GridBagConstraints.EAST;

gbc\_lblServiceBrake.insets = new Insets(0, 0, 5, 5);

gbc\_lblServiceBrake.gridx = 4;

gbc\_lblServiceBrake.gridy = 15;

theWindow.getContentPane().add(lblServiceBrake, gbc\_lblServiceBrake);

lblEngaged = new JLabel(bTout(sBrake));

GridBagConstraints gbc\_lblEngaged = new GridBagConstraints();

gbc\_lblEngaged.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged.insets = new Insets(0, 0, 5, 0);

gbc\_lblEngaged.gridx = 5;

gbc\_lblEngaged.gridy = 15;

theWindow.getContentPane().add(lblEngaged, gbc\_lblEngaged);

JButton btnTrackBlockNumber = new JButton("Track Block Number Override Toggle");

GridBagConstraints gbc\_btnTrackBlockNumber = new GridBagConstraints();

gbc\_btnTrackBlockNumber.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnTrackBlockNumber.insets = new Insets(0, 0, 5, 5);

gbc\_btnTrackBlockNumber.gridx = 0;

gbc\_btnTrackBlockNumber.gridy = 16;

theWindow.getContentPane().add(btnTrackBlockNumber, gbc\_btnTrackBlockNumber);

btnTrackBlockNumber.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

tbnOT=bToggle(tbnOT);

}

});

lblBlockOverride = new JLabel(bTout(tbnOT));

GridBagConstraints gbc\_lblBlockOverride = new GridBagConstraints();

gbc\_lblBlockOverride.anchor = GridBagConstraints.WEST;

gbc\_lblBlockOverride.insets = new Insets(0, 0, 5, 5);

gbc\_lblBlockOverride.gridx = 1;

gbc\_lblBlockOverride.gridy = 16;

theWindow.getContentPane().add(lblBlockOverride, gbc\_lblBlockOverride);

JButton btnServiceBrakeOverride = new JButton("Service Brake Override Toggle");

GridBagConstraints gbc\_btnServiceBrakeOverride = new GridBagConstraints();

gbc\_btnServiceBrakeOverride.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnServiceBrakeOverride.insets = new Insets(0, 0, 5, 5);

gbc\_btnServiceBrakeOverride.gridx = 4;

gbc\_btnServiceBrakeOverride.gridy = 16;

theWindow.getContentPane().add(btnServiceBrakeOverride, gbc\_btnServiceBrakeOverride);

btnServiceBrakeOverride.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

sBrakeOT=iToggle(sBrakeOT);

}

});

lblEngaged\_1 = new JLabel(iTbrake(sBrakeOT));

GridBagConstraints gbc\_lblEngaged\_1 = new GridBagConstraints();

gbc\_lblEngaged\_1.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged\_1.insets = new Insets(0, 0, 5, 0);

gbc\_lblEngaged\_1.gridx = 5;

gbc\_lblEngaged\_1.gridy = 16;

theWindow.getContentPane().add(lblEngaged\_1, gbc\_lblEngaged\_1);

JLabel lblCurrentIdealTemperature = new JLabel("Current Ideal Temperature (\u00B0C):");

GridBagConstraints gbc\_lblCurrentIdealTemperature = new GridBagConstraints();

gbc\_lblCurrentIdealTemperature.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblCurrentIdealTemperature.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentIdealTemperature.gridx = 0;

gbc\_lblCurrentIdealTemperature.gridy = 17;

theWindow.getContentPane().add(lblCurrentIdealTemperature, gbc\_lblCurrentIdealTemperature);

lblIdealtemp = new JLabel(""+cit);

GridBagConstraints gbc\_lblIdealtemp = new GridBagConstraints();

gbc\_lblIdealtemp.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblIdealtemp.insets = new Insets(0, 0, 5, 5);

gbc\_lblIdealtemp.gridx = 1;

gbc\_lblIdealtemp.gridy = 17;

theWindow.getContentPane().add(lblIdealtemp, gbc\_lblIdealtemp);

JLabel lblDoorsState = new JLabel("Doors State:");

GridBagConstraints gbc\_lblDoorsState = new GridBagConstraints();

gbc\_lblDoorsState.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblDoorsState.insets = new Insets(0, 0, 5, 5);

gbc\_lblDoorsState.gridx = 4;

gbc\_lblDoorsState.gridy = 17;

theWindow.getContentPane().add(lblDoorsState, gbc\_lblDoorsState);

lblClosed = new JLabel(oDoors(doors));

GridBagConstraints gbc\_lblClosed = new GridBagConstraints();

gbc\_lblClosed.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblClosed.insets = new Insets(0, 0, 5, 0);

gbc\_lblClosed.gridx = 5;

gbc\_lblClosed.gridy = 17;

theWindow.getContentPane().add(lblClosed, gbc\_lblClosed);

JLabel lblIdealTemperaturec = new JLabel("Ideal Temperature (\u00B0C):");

GridBagConstraints gbc\_lblIdealTemperaturec = new GridBagConstraints();

gbc\_lblIdealTemperaturec.anchor = GridBagConstraints.EAST;

gbc\_lblIdealTemperaturec.insets = new Insets(0, 0, 5, 5);

gbc\_lblIdealTemperaturec.gridx = 0;

gbc\_lblIdealTemperaturec.gridy = 18;

theWindow.getContentPane().add(lblIdealTemperaturec, gbc\_lblIdealTemperaturec);

txtUser\_7 = new JTextField();

txtUser\_7.setText("user #");

GridBagConstraints gbc\_txtUser\_7 = new GridBagConstraints();

gbc\_txtUser\_7.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_7.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_7.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_7.gridx = 1;

gbc\_txtUser\_7.gridy = 18;

theWindow.getContentPane().add(txtUser\_7, gbc\_txtUser\_7);

txtUser\_7.setColumns(10);

JButton btnSet\_7 = new JButton("set");

GridBagConstraints gbc\_btnSet\_7 = new GridBagConstraints();

gbc\_btnSet\_7.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_7.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_7.gridx = 2;

gbc\_btnSet\_7.gridy = 18;

theWindow.getContentPane().add(btnSet\_7, gbc\_btnSet\_7);

btnSet\_7.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

double currDou=Double.parseDouble(txtUser\_7.getText());

cit=currDou;

}

});

JLabel lblCurrentAnnouncement = new JLabel("Current Announcement:");

GridBagConstraints gbc\_lblCurrentAnnouncement = new GridBagConstraints();

gbc\_lblCurrentAnnouncement.anchor = GridBagConstraints.EAST;

gbc\_lblCurrentAnnouncement.insets = new Insets(0, 0, 5, 5);

gbc\_lblCurrentAnnouncement.gridx = 4;

gbc\_lblCurrentAnnouncement.gridy = 18;

theWindow.getContentPane().add(lblCurrentAnnouncement, gbc\_lblCurrentAnnouncement);

lblTheRainIn = new JLabel(announ);

GridBagConstraints gbc\_lblTheRainIn = new GridBagConstraints();

gbc\_lblTheRainIn.anchor = GridBagConstraints.WEST;

gbc\_lblTheRainIn.insets = new Insets(0, 0, 5, 0);

gbc\_lblTheRainIn.gridx = 5;

gbc\_lblTheRainIn.gridy = 18;

theWindow.getContentPane().add(lblTheRainIn, gbc\_lblTheRainIn);

JLabel lblTimemilitary = new JLabel("Time (military):");

GridBagConstraints gbc\_lblTimemilitary = new GridBagConstraints();

gbc\_lblTimemilitary.anchor = GridBagConstraints.NORTHEAST;

gbc\_lblTimemilitary.insets = new Insets(0, 0, 5, 5);

gbc\_lblTimemilitary.gridx = 0;

gbc\_lblTimemilitary.gridy = 19;

theWindow.getContentPane().add(lblTimemilitary, gbc\_lblTimemilitary);

lblTime = new JLabel(timeH+":"+timeM+":"+timeS);

GridBagConstraints gbc\_lblTime = new GridBagConstraints();

gbc\_lblTime.anchor = GridBagConstraints.NORTHWEST;

gbc\_lblTime.insets = new Insets(0, 0, 5, 5);

gbc\_lblTime.gridx = 1;

gbc\_lblTime.gridy = 19;

theWindow.getContentPane().add(lblTime, gbc\_lblTime);

JLabel lblLightsStatus = new JLabel("Lights Status:");

GridBagConstraints gbc\_lblLightsStatus = new GridBagConstraints();

gbc\_lblLightsStatus.anchor = GridBagConstraints.EAST;

gbc\_lblLightsStatus.insets = new Insets(0, 0, 5, 5);

gbc\_lblLightsStatus.gridx = 4;

gbc\_lblLightsStatus.gridy = 19;

theWindow.getContentPane().add(lblLightsStatus, gbc\_lblLightsStatus);

lblLightsOn = new JLabel(oLights(lights));

GridBagConstraints gbc\_lblLightsOn = new GridBagConstraints();

gbc\_lblLightsOn.anchor = GridBagConstraints.WEST;

gbc\_lblLightsOn.insets = new Insets(0, 0, 5, 0);

gbc\_lblLightsOn.gridx = 5;

gbc\_lblLightsOn.gridy = 19;

theWindow.getContentPane().add(lblLightsOn, gbc\_lblLightsOn);

JLabel lblTimeOverridehours = new JLabel("Time Override (military, hours):");

GridBagConstraints gbc\_lblTimeOverridehours = new GridBagConstraints();

gbc\_lblTimeOverridehours.anchor = GridBagConstraints.EAST;

gbc\_lblTimeOverridehours.insets = new Insets(0, 0, 5, 5);

gbc\_lblTimeOverridehours.gridx = 0;

gbc\_lblTimeOverridehours.gridy = 20;

theWindow.getContentPane().add(lblTimeOverridehours, gbc\_lblTimeOverridehours);

txtUser\_8 = new JTextField();

txtUser\_8.setText("user #");

GridBagConstraints gbc\_txtUser\_8 = new GridBagConstraints();

gbc\_txtUser\_8.anchor = GridBagConstraints.NORTH;

gbc\_txtUser\_8.fill = GridBagConstraints.HORIZONTAL;

gbc\_txtUser\_8.insets = new Insets(0, 0, 5, 5);

gbc\_txtUser\_8.gridx = 1;

gbc\_txtUser\_8.gridy = 20;

theWindow.getContentPane().add(txtUser\_8, gbc\_txtUser\_8);

txtUser\_8.setColumns(10);

JButton btnSet\_8 = new JButton("set");

GridBagConstraints gbc\_btnSet\_8 = new GridBagConstraints();

gbc\_btnSet\_8.anchor = GridBagConstraints.SOUTHWEST;

gbc\_btnSet\_8.insets = new Insets(0, 0, 5, 5);

gbc\_btnSet\_8.gridx = 2;

gbc\_btnSet\_8.gridy = 20;

theWindow.getContentPane().add(btnSet\_8, gbc\_btnSet\_8);

btnSet\_8.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

double currDou=Double.parseDouble(txtUser\_8.getText());

timeO=currDou;

}

});

JLabel lblEbrakeState = new JLabel("E-Brake State:");

GridBagConstraints gbc\_lblEbrakeState = new GridBagConstraints();

gbc\_lblEbrakeState.anchor = GridBagConstraints.EAST;

gbc\_lblEbrakeState.insets = new Insets(0, 0, 5, 5);

gbc\_lblEbrakeState.gridx = 4;

gbc\_lblEbrakeState.gridy = 20;

theWindow.getContentPane().add(lblEbrakeState, gbc\_lblEbrakeState);

lblEngaged\_2 = new JLabel(bTout(eBrake));

GridBagConstraints gbc\_lblEngaged\_2 = new GridBagConstraints();

gbc\_lblEngaged\_2.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged\_2.insets = new Insets(0, 0, 5, 0);

gbc\_lblEngaged\_2.gridx = 5;

gbc\_lblEngaged\_2.gridy = 20;

theWindow.getContentPane().add(lblEngaged\_2, gbc\_lblEngaged\_2);

JButton btnTimeOverrideToggle = new JButton("Time Override Toggle");

GridBagConstraints gbc\_btnTimeOverrideToggle = new GridBagConstraints();

gbc\_btnTimeOverrideToggle.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnTimeOverrideToggle.insets = new Insets(0, 0, 0, 5);

gbc\_btnTimeOverrideToggle.gridx = 0;

gbc\_btnTimeOverrideToggle.gridy = 21;

theWindow.getContentPane().add(btnTimeOverrideToggle, gbc\_btnTimeOverrideToggle);

btnTimeOverrideToggle.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

timeOT=bToggle(timeOT);

}

});

lblTimeOverride = new JLabel(bTout(timeOT));

GridBagConstraints gbc\_lblTimeOverride = new GridBagConstraints();

gbc\_lblTimeOverride.anchor = GridBagConstraints.WEST;

gbc\_lblTimeOverride.insets = new Insets(0, 0, 0, 5);

gbc\_lblTimeOverride.gridx = 1;

gbc\_lblTimeOverride.gridy = 21;

theWindow.getContentPane().add(lblTimeOverride, gbc\_lblTimeOverride);

JButton btnEbrakeOverrideToggle = new JButton("E-Brake Override Toggle");

GridBagConstraints gbc\_btnEbrakeOverrideToggle = new GridBagConstraints();

gbc\_btnEbrakeOverrideToggle.anchor = GridBagConstraints.NORTHEAST;

gbc\_btnEbrakeOverrideToggle.insets = new Insets(0, 0, 0, 5);

gbc\_btnEbrakeOverrideToggle.gridx = 4;

gbc\_btnEbrakeOverrideToggle.gridy = 21;

theWindow.getContentPane().add(btnEbrakeOverrideToggle, gbc\_btnEbrakeOverrideToggle);

btnEbrakeOverrideToggle.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e)

{

//Execute when button is pressed

eBrakeOT=iToggle(eBrakeOT);

}

});

lblEngaged\_3 = new JLabel(iTfail(eBrakeOT));

GridBagConstraints gbc\_lblEngaged\_3 = new GridBagConstraints();

gbc\_lblEngaged\_3.anchor = GridBagConstraints.WEST;

gbc\_lblEngaged\_3.gridx = 5;

gbc\_lblEngaged\_3.gridy = 21;

theWindow.getContentPane().add(lblEngaged\_3, gbc\_lblEngaged\_3);

theWindow.setVisible(true);

}

// toggle for engaged/disengaged options

private boolean bToggle(boolean currboo)

{

if(currboo)

{

currboo=false;

}

else

{

currboo=true;

}

return currboo;

}

// String builder for boolean toggles

private String bTout(boolean currboo)

{

String outs;

if(currboo)

{

outs="Engaged";

}

else

{

outs="Disengaged";

}

return outs;

}

// String builder for failure modes

private String bFout(boolean currboo)

{

String outs;

if(currboo)

{

outs="Failure";

}

else

{

outs="No Failure";

}

return outs;

}

// creates the string for the pause state

private String bPause(boolean currboo)

{

String outs;

if(currboo)

{

outs="Paused";

}

else

{

outs="Running";

}

return outs;

}

// creates the string for the tick control

private String bTick(boolean currboo)

{

String outs;

if(currboo)

{

outs="Test Control";

}

else

{

outs="CTC Control";

}

return outs;

}

// the toggle method for failure modes and brakes

private int iToggle(int currint)

{

if(currint<2)

{

currint++;

}

else

{

currint=0;

}

return currint;

}

//creates the string for the failure overrides

private String iTfail(int currint)

{

String outs;

if(currint==0)

{

outs="Disengaged";

}

else if(currint==1)

{

outs="Forced Fail";

}

else

{

outs="Forced Pass";

}

return outs;

}

// creates the string for the brake overrides

private String iTbrake(int currint)

{

String outs;

if(currint==0)

{

outs="Override Off";

}

else if(currint==1)

{

outs="Forced Brake";

}

else

{

outs="Forced No Brake";

}

return outs;

}

// creates the string for the doors status

private String oDoors(boolean currboo)

{

String outs;

if(currboo)

{

outs="Open";

}

else

{

outs="Closed";

}

return outs;

}

// creates the string for the lights status

private String oLights(boolean currboo)

{

String outs;

if(currboo)

{

outs="On";

}

else

{

outs="Off";

}

return outs;

}

// divides the time into seconds, minutes, and hours

private void timeCalc(double ntime)

{

timeS=(int)(ntime%60);

timeM=(int)((ntime/60)%60);

timeH=(int)(ntime/3600);

}

// creates the string for the time

private String timeString()

{

if(timeS<10)

{

if(timeM<10)

{

return timeH+":0"+timeM+":0"+timeS;

}

else

{

return timeH+":"+timeM+":0"+timeS;

}

}

else if(timeM<10)

{

return timeH+":0"+timeM+":"+timeS;

}

else

{

return timeH+":"+timeM+":"+timeS;

}

}

// updates UI

public ResponseUI updateUI(TrainController uiTNC)

{

//do not update if paused

if(!pause)

{

// take values from TNC

currVel=Math.round(uiTNC.currspeed\*1000)/1000;

currPow=Math.round(uiTNC.power\*1000)/1000;

speedLimit=uiTNC.slimit;

aut=Math.round(uiTNC.autspeed\*1000)/1000;

plcVel=Math.round(uiTNC.safespeed\*1000)/1000;

tbn=uiTNC.thisBlock.id;

timeCalc(uiTNC.time);

trainEF=uiTNC.engineFail;

signalPF=uiTNC.signalFail;

brakeF=uiTNC.brakeFail;

brokenR=uiTNC.railFail;

trackCF=uiTNC.trackFail;

powerF=uiTNC.powerFail;

sBrake=uiTNC.sBrake;

eBrake=uiTNC.eBrake;

doors=uiTNC.doors;

announ=uiTNC.currAnnoun;

lights=uiTNC.lights;

//update UI labels

lblVel.setText(""+currVel);

label.setText(""+currTrain);

lblPause.setText(bPause(pause));

lblEngaged\_CVOT.setText(bTout(cvoT));

lblFail.setText(bFout(trainEF));

lblPow.setText(""+currPow);

lblFail\_1.setText(iTfail(trainEFoT));

lblSetVel.setText(""+currSetpoint);

lblFail\_2.setText(bFout(signalPF));

lblFail\_3.setText(iTfail(signalPFoT));

lblVelLimit.setText(""+speedLimit);

lblFail\_4.setText(bFout(brakeF));

lblFail\_5.setText(iTfail(brakeFoT));

lblEngaged\_5.setText(bTout(sloT));

lblFail\_6.setText(bFout(brokenR));

lblAutVel.setText(""+aut);

lblFail\_7.setText(iTfail(brokenRoT));

lblFail\_8.setText(bFout(trackCF));

lblEngaged\_6.setText(bTout(autOT));

lblFail\_9.setText(iTfail(trackCFoT));

lblPlcVel.setText(""+plcVel);

lblFail\_10.setText(bFout(powerF));

lblBlock.setText(""+tbn);

lblFail\_11.setText(iTfail(powerFoT));

lblEngaged.setText(bTout(sBrake));

lblBlockOverride.setText(bTout(tbnOT));

lblEngaged\_1.setText(iTbrake(sBrakeOT));

lblIdealtemp.setText(""+cit);

lblClosed.setText(oDoors(doors));

lblTheRainIn.setText(announ);

lblTime.setText(timeString());

lblLightsOn.setText(oLights(lights));

lblEngaged\_2.setText(bTout(eBrake));

lblTimeOverride.setText(bTout(timeOT));

lblEngaged\_3.setText(iTfail(eBrakeOT));

//send response to Train Controller

ResponseUI tncRes=new ResponseUI(cvo, cvoT, currSetpoint, slo, sloT, autO, autOT, tbnO, tbnOT,

timeO, timeOT, trainEFoT, signalPFoT, brakeFoT, brokenRoT, trackCFoT, powerFoT, sBrakeOT,

eBrakeOT, cit, false);

return tncRes;

}

else

{

//send paused response to Train Controller if Paused.

ResponseUI tncRes=new ResponseUI(cvo, cvoT, currSetpoint, slo, sloT, autO, autOT, tbnO, tbnOT,

timeO, timeOT, trainEFoT, signalPFoT, brakeFoT, brokenRoT, trackCFoT, powerFoT, sBrakeOT,

eBrakeOT, cit, true);

return tncRes;

}

}

}