```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package seafarm;
import de.re.easymodbus.modbusclient.*;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.util.Arrays;
import java.util.logging.Level;
import java.util.logging.Logger;
/**
 *
 * @author sigurdolav
 */
public class ModbusReciever implements Runnable{

    //modbus client, set to the same client as the sender in constructor//
    private ModbusClient client;
    private Thread t;
    private RecieveDataObserver observer;
    private boolean connected=false;
    //data to recieve from platform///
    private float platLat=0;
    private float platLong=0;
    private float currentSpeed=0;
    private float pitch=0;
    private float yaw=0;
    private float roll=0;
    private int thrusterPower=0;
    private boolean enabled=false;
    private boolean rovLocked=false;
    private boolean rovUpperPos=false;
    private boolean dpMode = false;
    private boolean autoMode = false;
    private boolean manualMode = false;


    //the register adress of the different variables on the PLC//
    //heading forandres til pitch
    private final int yawReg=4;
```

```
    private final int rollReg=8;
    private final int enabledReg=192;
    private final int pitchReg=14;
    private final int currenSpeedReg=12;
    private final int platLongReg=0;
    private final int platLatReg=28;
    private final int rovLockedReg=384;
    private final int rovUpperPosReg=400;
    private final int dpModeReg = 416;
    private final int autoModeReg = 387;
    private final int manualModeReg = 388;


    private int readFrequency=1; //2 times a secound if var =2

    public ModbusReciever(ModbusClient modbusClient,RecieveDataObserver
obs){

        this.client=modbusClient;
        this.observer=obs;


    }

     /**
     * create thread and starts it
     */
    public void start() {
        t = new Thread(this, "MODBUSreceiverThread");
        t.start();
        System.out.println("Starting");
    }

        /**
     * read the registers on the PLC and update the recieve data observer
     */
    public void run() {
        try {
            Thread.sleep(8000);
        } catch (InterruptedException ex) {
            Logger.getLogger(ModbusReciever.class.getName()).log(Level.SEVE
RE, null, ex);
        }
        if (observer != null) {
```

```
            while(!connected){
            try {
            System.out.println("Modbus recieverTrying to connect to
server");
            client.Connect();
            connected=true;
                }
            catch (Exception e) {
                connected=false;
                System.out.println("connecting exception Modbus reciever");
                continue;

            }
            }


            try{
                while (observer.shouldChildOfThisRun()) {

                    recieveDataFromServer();
                    //System.out.println("recieving data from server");
                    updateRecieveObserver();
                    //System.out.println("updatingRecieverObserver");
                    //System.out.println("modbus has been recieved");
                    Thread.sleep(1000/this.readFrequency);

                     }
                }
            catch (Exception e)
            {
                connected=false;
                System.out.println("exception reciev modbus under update
recieve"+e);
            }
        }

        else {
            System.out.println("receive datahandler not created in
udpreceiver thread");
        }
        try {
            client.Disconnect();
```

```
    }

    catch (Exception e)
    {
        System.out.println("modbustcp.ModbusReciever.disconnect()");
    }

}

private void recieveDataFromServer(){

   // System.out.println("Trying to read from plc");
    this.pitch=ReadFloat(pitchReg);
    this.platLat=ReadFloat(platLatReg);
    this.platLong=ReadFloat(platLongReg);
    //System.out.println(this.platLong);
    //System.out.println(this.platLat);

    this.roll = ReadFloat(rollReg);
    this.currentSpeed=ReadFloat(currenSpeedReg);
    this.yaw=ReadFloat(yawReg);
    this.enabled=ReadBool(enabledReg);
    this.rovLocked=ReadBool(rovLockedReg);
    this.rovUpperPos=ReadBool(rovUpperPosReg);
    this.dpMode = ReadBool(dpModeReg);
    this.autoMode = ReadBool(autoModeReg);
    this.manualMode = ReadBool(manualModeReg);
    // System.out.println("finsished RecieveDataFrom Server");

}

private void updateRecieveObserver(){


    observer.setCurrentPitch(this.pitch);

    observer.setCurrentLatitude(this.platLat);
    observer.setCurrentLongitude(this.platLong);
    observer.setCurrentRoll(this.roll);
    observer.setCurrentSpeed(this.currentSpeed);
    observer.setCurrentYaw(this.yaw);
    observer.setEnabled(this.enabled);
    observer.setRovLock(this.rovLocked);
    observer.setRovUpperPos(this.rovUpperPos);
```

```
        observer.setDpMode(this.dpMode);
        observer.setAutoMode(this.autoMode);
        observer.setManualMode(this.manualMode);

        observer.notifyObs();
        //System.out.println("observer call ");


    }
    //example         //Read Float Value from Register 10 and 11
                        //System.out.println(ModbusClient.ConvertRegistersT
oFloat(modbusClient.ReadHoldingRegisters(9, 2)));
    private float ReadFloat(int reg){
        int regval=reg;
        float val=0;
        try
        {
        val=
this.client.ConvertRegistersToFloat(this.client.ReadHoldingRegisters(regval
, 2));

        }
        catch(Exception e)
        {
            System.out.println("modbustcp.ModbusReciever.ReadFloat()");
        }
        return val;
    }

    private boolean ReadBool(int reg){
       boolean retVal=false;
       boolean[]  val;
        try {
         val=client.ReadCoils(reg, 1);
         retVal= val[0];
        } catch (Exception e)
        {
            System.out.println("modbustcp.ModbusReciever.ReadBool()");
            connected=false;
            run();

        }
        return retVal;
```

```
    }

}
```