Keypad

Om bij de automaat de pincode te kunnen intoetsen gebruiken we een 4x4 keypad.

Vanuit je pinautomaat de pincode te ontvangen gebruiken we de Arduino.

voorbeeld:

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * Automaat.java
 *
 * Created on Oct 8, 2013, 10:13:46 AM
 */

package javaarduino;

import gnu.io.CommPortIdentifier;
import gnu.io.SerialPort;
import gnu.io.SerialPortEvent;
import gnu.io.SerialPortEventListener;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.util.Enumeration;

/**
 * gno.io.* komt van rxtxcomm.jar
 * @author hroblajf
 */
public class Automaat extends javax.swing.JFrame implements SerialPortEventListener {

    /** Creates new form Automaat */
    SerialPort serialPort;
        Automaat automaat=null;
        /** The port we're normally going to use. */
            private static final String PORT_NAMES[] = {
                                    "/dev/tty.usbmodem1d11", // Mac OS X
                                    "/dev/ttyUSB0", // Linux
                                    "COM3", // Windows
            };
            /**
            * A BufferedReader which will be fed by a InputStreamReader
            * converting the bytes into characters
            * making the displayed results codepage independent
            */
            private BufferedReader input;
            /** The output stream to the port */
            private OutputStream output;
            /** Milliseconds to block while waiting for port open */
            private static final int TIME_OUT = 2000;
            /** Default bits per second for COM port. */
```

```
        private static final int DATA_RATE = 9600;

public void initialize() {
                CommPortIdentifier portId = null;
                Enumeration portEnum = CommPortIdentifier.getPortIdentifiers();

                //First, Find an instance of serial port as set in PORT_NAMES.
                while (portEnum.hasMoreElements()) {
                        CommPortIdentifier currPortId = (CommPortIdentifier) portEnum.nextElement();
                        for (String portName : PORT_NAMES) {
                                if (currPortId.getName().equals(portName)) {
                                        portId = currPortId;
                                        break;
                                }
                        }
                }
                if (portId == null) {
                        System.out.println("Could not find COM port.");
                        return;
                }

                try {
                        // open serial port, and use class name for the appName.
                        serialPort = (SerialPort) portId.open(this.getClass().getName(),
                                        TIME_OUT);

                        // set port parameters
                        serialPort.setSerialPortParams(DATA_RATE,
                                        SerialPort.DATABITS_8,
                                        SerialPort.STOPBITS_1,
                                        SerialPort.PARITY_NONE);

                        // open the streams
                        input = new BufferedReader(new InputStreamReader(serialPort.getInputStream()));
                        output = serialPort.getOutputStream();

                        // add event listeners
                        serialPort.addEventListener(this);
                        serialPort.notifyOnDataAvailable(true);
                } catch (Exception e) {
                        System.err.println(e.toString());
                }
        }


public Automaat() {
   initComponents();
   jTextField1.setEditable(false);
   initialize();
                Thread t=new Thread() {
                        public void run() {
        //the following line will keep this app alive for 10 seconds,
        //waiting for events to occur and responding to them (printing incoming messages to console).
                                try {Thread.sleep(10000);} catch (InterruptedException ie) {}
                  System.exit(0);
                        }
                };
                t.start();
                System.out.println("Started");


}
```

```
/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jTextField1 = new javax.swing.JTextField();
    jLabel1 = new javax.swing.JLabel();
    jButton1 = new javax.swing.JButton();
    jButton2 = new javax.swing.JButton();
    jButton3 = new javax.swing.JButton();
    jLabel2 = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

    jTextField1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jTextField1ActionPerformed(evt);
        }
    });

    jLabel1.setText("Toets uw pincode in");

    jButton1.setText("back");
    jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
        }
    });

    jButton2.setText("CE");
    jButton2.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton2ActionPerformed(evt);
        }
    });

    jButton3.setText("OK");
    jButton3.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton3ActionPerformed(evt);
        }
    });

    org.jdesktop.layout.GroupLayout layout = new org.jdesktop.layout.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
        .add(layout.createSequentialGroup()
            .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                .add(layout.createSequentialGroup()
                    .add(69, 69, 69)
                    .add(jLabel1))
                .add(layout.createSequentialGroup()
                    .add(136, 136, 136)
                    .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING, false)
                        .add(layout.createSequentialGroup()
```

```
                        .add(jButton1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 79,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(jButton2, 0, 0, Short.MAX_VALUE))
                    .add(jTextField1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 161, Short.MAX_VALUE)
                    .add(jButton3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                .add(layout.createSequentialGroup()
                    .add(96, 96, 96)
                    .add(jLabel2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 238,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
            .addContainerGap(66, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(18, 18, 18)
                .add(jLabel1)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jTextField1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jButton1)
                    .add(jButton2))
                .add(18, 18, 18)
                .add(jButton3)
                .add(72, 72, 72)
                .add(jLabel2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 17,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(58, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String nu = jTextField1.getText();
        if (nu.length()>0){
            nu = nu.substring(0,nu.length()-1);
        }
        jTextField1.setText(nu);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        jTextField1.setText("");
        jLabel2.setText("");
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        jLabel2.setText(jTextField1.getText());
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
```

```
        public void run() {
            new Automaat().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JTextField jTextField1;
// End of variables declaration

public void serialEvent(SerialPortEvent spe) {
  if (spe.getEventType() == SerialPortEvent.DATA_AVAILABLE) {
                        try {
                            String inputLine=input.readLine();
                if (inputLine.equals("#")) jTextField1.setText("");
                else{
                jTextField1.setText(jTextField1.getText()+inputLine);
                if (jTextField1.getText().length()>4) jTextField1.setText(inputLine);
                            //System.out.println(inputLine);
                }
                        } catch (Exception e) {
                            System.err.println(e.toString());
                        }
                }
}

}
```

Dit programma vangt nu de toetsaanslagen van het keypad af.

Op de arduino draait een programma met de Keypad library van de arduino.
Toetsaanslagen worden met Serial.println(….) verstuurd.

```
#include <Keypad.h>
int count = 0;
const byte ROWS = 4; // Four rows
const byte COLS = 4; // Three columns
// Define the Keymap
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
// Connect keypad ROW0, ROW1, ROW2 and ROW3 to these Arduino pins.
byte rowPins[ROWS] = { 5, 6, 7, 8 };
// Connect keypad COL0, COL1 and COL2 to these Arduino pins.
byte colPins[COLS] = { 9, 10, 11, 12 };

// Create the Keypad
Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

#define ledpin 13

void setup()
{
```

```
  pinMode(ledpin,OUTPUT);
  digitalWrite(ledpin, HIGH);
  Serial.begin(9600);
}

void loop()
{
  char key = kpd.getKey();
  if(key)  // Check for a valid key.
  {

    Serial.println(key);

  }
}
```