Codigo

#include <Ultrasonic.h>

/\* Pulse Sensor Amped 1.4 by Joel Murphy and Yury Gitman http://www.pulsesensor.com

---------------------- Notes ---------------------- ----------------------

This code:

1) Blinks an LED to User's Live Heartbeat PIN 13

2) Fades an LED to User's Live HeartBeat

3) Determines BPM

4) Prints All of the Above to Serial

Read Me:

https://github.com/WorldFamousElectronics/PulseSensor\_Amped\_Arduino/blob/master/README.md

---------------------- ---------------------- ----------------------

\*/

// Variables

float sensor =0;

int option;

int pulsacion=0;

int pulsePin = 0; // Pulse Sensor purple wire connected to analog pin 0

int blinkPin = 13; // pin to blink led at each beat

int fadePin = 5; // pin to do fancy classy fading blink at each beat

int fadeRate = 0; // used to fade LED on with PWM on fadePin

int contador=0;

// Volatile Variables, used in the interrupt service routine!

volatile int BPM; // int that holds raw Analog in 0. updated every 2mS

volatile int Signal; // holds the incoming raw data

volatile int IBI = 600; // int that holds the time interval between beats! Must be seeded!

volatile boolean Pulse = false; // "True" when User's live heartbeat is detected. "False" when not a "live beat".

volatile boolean QS = false; // becomes true when Arduoino finds a beat.

// Regards Serial OutPut -- Set This Up to your needs

static boolean serialVisual = true; // Set to 'false' by Default. Re-set to 'true' to see Arduino Serial Monitor ASCII Visual Pulse

Ultrasonic ultrasonic(6,7,23200);// (Trig PIN,Echo PIN, TIMEOUT)

//TIMEOUT = (CENTIMETROS)\*(58)

void setup(){

pinMode(blinkPin,OUTPUT); // pin that will blink to your heartbeat!

pinMode(fadePin,OUTPUT); // pin that will fade to your heartbeat!

//Serial.begin(115200); // we agree to talk fast!

Serial.begin(9600); // we agree to talk fast!

interruptSetup(); // sets up to read Pulse Sensor signal every 2mS

// IF YOU ARE POWERING The Pulse Sensor AT VOLTAGE LESS THAN THE BOARD VOLTAGE,

// UN-COMMENT THE NEXT LINE AND APPLY THAT VOLTAGE TO THE A-REF PIN

// analogReference(EXTERNAL);

}

// Where the Magic Happens

void loop()

{

serialOutput() ;

if (QS == true)

{ // A Heartbeat Was Found

// BPM and IBI have been Determined

// Quantified Self "QS" true when arduino finds a heartbeat

fadeRate = 255; // Makes the LED Fade Effect Happen

// Set 'fadeRate' Variable to 255 to fade LED with pulse

serialOutputWhenBeatHappens(); // A Beat Happened, Output that to serial.

QS = false; // reset the Quantified Self flag for next time

}

ledFadeToBeat(); // Makes the LED Fade Effect Happen

delay(20); // take a break

}

void ledFadeToBeat()

{

fadeRate -= 15; // set LED fade value

fadeRate = constrain(fadeRate,0,255); // keep LED fade value from going into negative numbers!

analogWrite(fadePin,fadeRate); // fade LED

if ((BPM>90)&(BPM<150))

{

pulsacion=BPM;

}

if (Serial.available()>0)

{

option=Serial.read();

}

if(option=='1')

{

sensor =(analogRead(A1)\*48875)/100000;

Serial.println(pulsacion);

Serial.println(ultrasonic.Ranging(CM)); // CM or INC

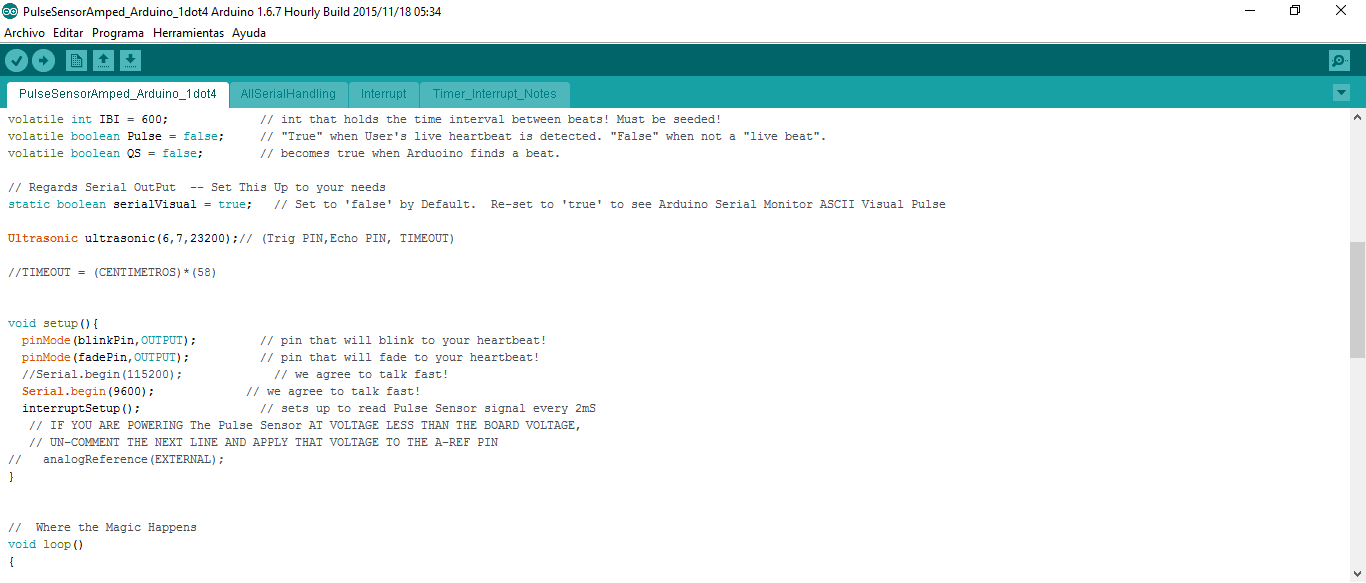
Serial.println(sensor);

option=0;

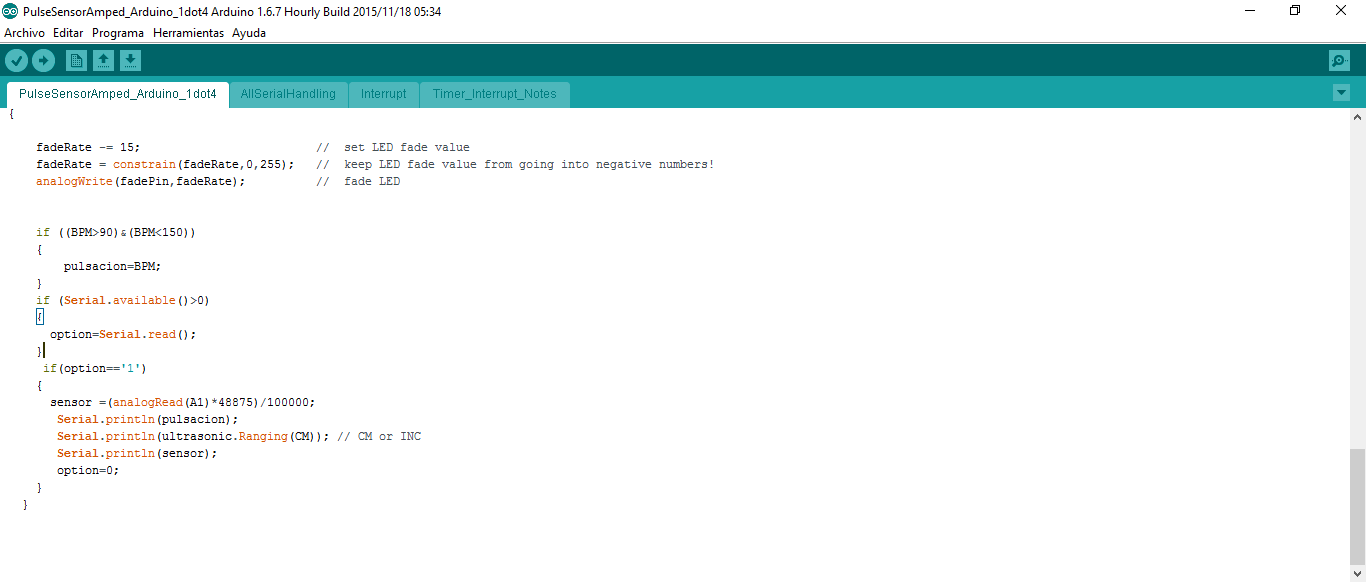
}

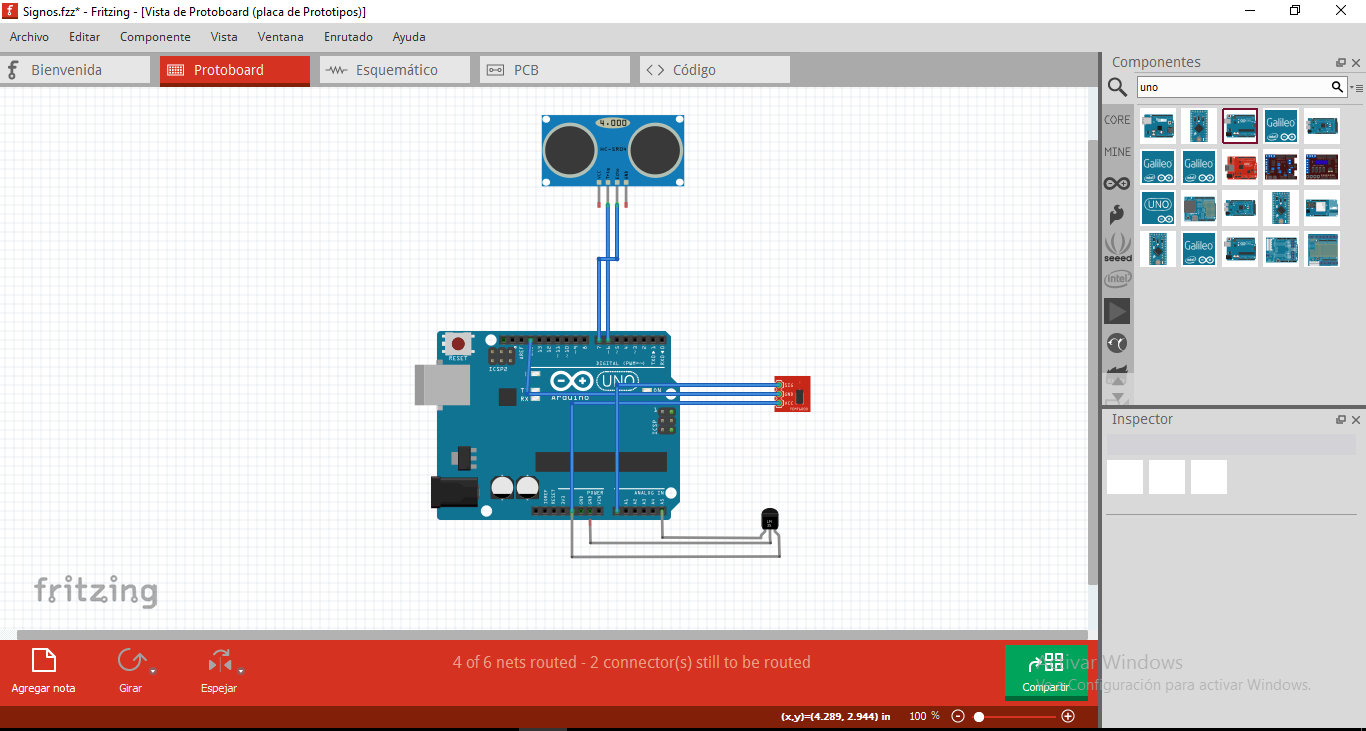
}

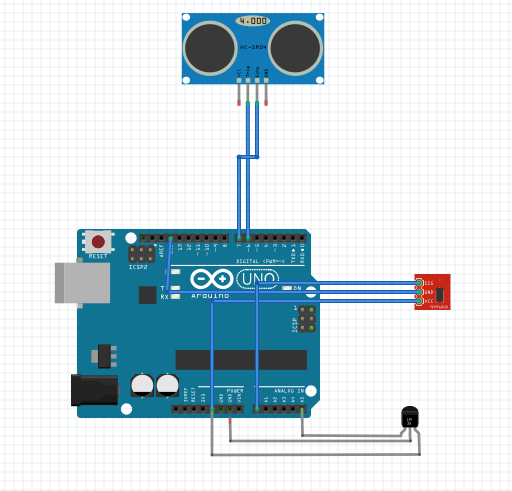




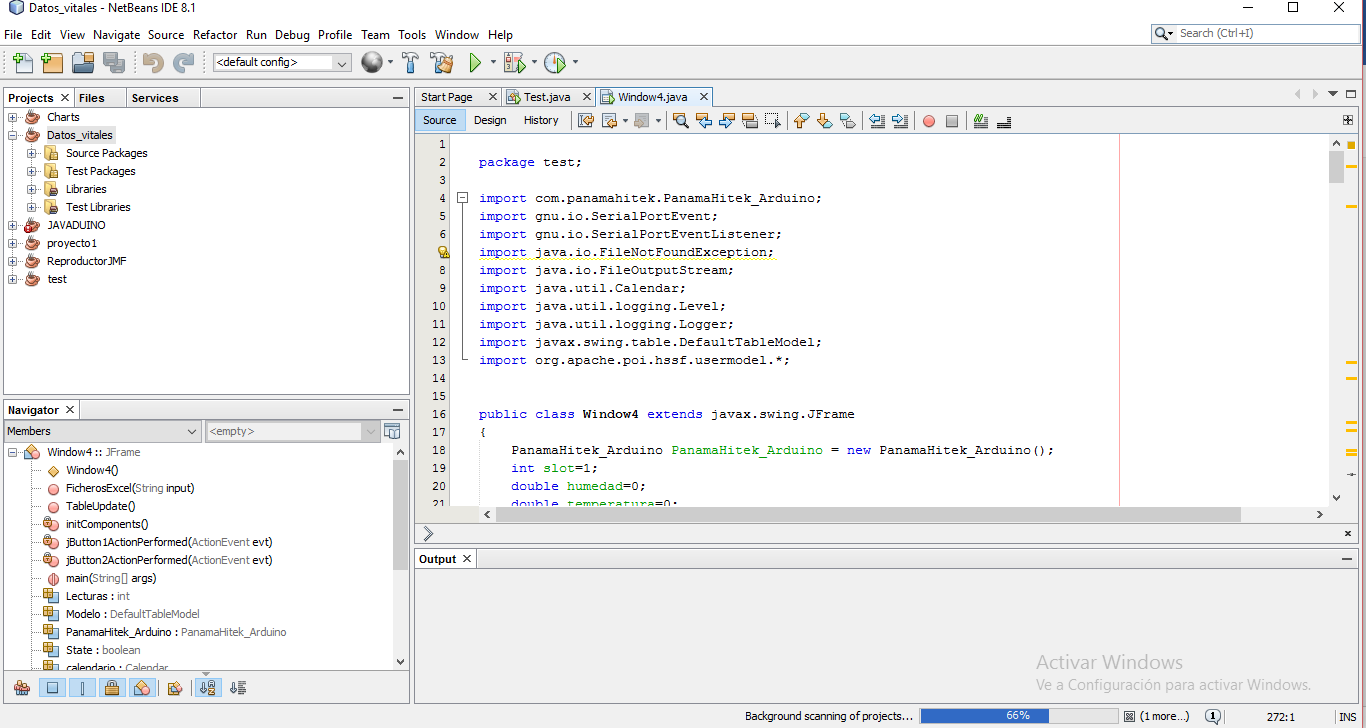








**Diseño software Java Netbeans**



**Codigo**

package test;

import com.panamahitek.PanamaHitek\_Arduino;

import gnu.io.SerialPortEvent;

import gnu.io.SerialPortEventListener;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.util.Calendar;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.table.DefaultTableModel;

import org.apache.poi.hssf.usermodel.\*;

public class Window4 extends javax.swing.JFrame

{

PanamaHitek\_Arduino PanamaHitek\_Arduino = new PanamaHitek\_Arduino();

int slot=1;

double humedad=0;

double temperatura=0;

double temperatura2=0;//john

int Lecturas =0;

Calendar calendario;

SerialPortEventListener evento = new SerialPortEventListener() {

@Override

public void serialEvent(SerialPortEvent spe)

{

if (PanamaHitek\_Arduino.MessageAvailable())

{

//System.out.println(PanamaHitek\_Arduino.printMessage());

if (slot==1)

{

if(Lecturas>1)

{

TableUpdate();

}

slot=2;

Lecturas++;

humedad = Double.parseDouble(PanamaHitek\_Arduino.printMessage());

}

else if (slot==2)

{

slot=3;

Lecturas++;

temperatura = Double.parseDouble(PanamaHitek\_Arduino.printMessage());

}

//john

else if (slot==3)

{

slot=1;

Lecturas++;

temperatura2 = Double.parseDouble(PanamaHitek\_Arduino.printMessage());

}

}

}

};

DefaultTableModel Modelo;

boolean State=false;

public void TableUpdate()

{

String Output="";

int hora = calendario.get(Calendar.HOUR\_OF\_DAY);

int minuto = calendario.get(Calendar.MINUTE);

int segundo = calendario.get(Calendar.SECOND);

if (hora<10)

Output= "0"+hora+":"+minuto+":"+segundo;

else if(minuto<10)

Output= hora+":"+"0"+minuto+":"+segundo;

else if(segundo<10)

Output= hora+":"+minuto+":"+"0"+segundo;

else

Output= hora+":"+minuto+":"+segundo;

calendario = Calendar.getInstance();

//System.out.println("Temperatura: "+temperatura+" Humedad: "+humedad);

//Modelo.addRow(new Object[]{""+Output,humedad,temperatura});

Modelo.addRow(new Object[]{humedad,temperatura,temperatura2});

}

public Window4()

{

this.calendario = Calendar.getInstance();

initComponents();

Modelo = (DefaultTableModel) jTable1.getModel();

try

{

PanamaHitek\_Arduino.ArduinoRXTX("COM9", 2000, 9600, evento);

} catch (Exception ex)

{

Logger.getLogger(Window4.class.getName()).log(Level.SEVERE, null, ex);

}

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Ritmo Cardiaco", "Altura", "Temperatura"

}

));

jScrollPane1.setViewportView(jTable1);

jButton1.setText("Iniciar toma de datos");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton2.setText("Exportar a Excel");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap(15, Short.MAX\_VALUE)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 375, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

.addGroup(layout.createSequentialGroup()

.addGap(48, 48, 48)

.addComponent(jButton1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton2)

.addGap(70, 70, 70))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 203, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton1)

.addComponent(jButton2))

.addContainerGap(31, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

if (State==true)

{

jButton1.setText("Iniciar toma de datos");

State=false;

try {

//Modelo.addRow(new Object[]{"1","2","3"});

PanamaHitek\_Arduino.sendData("1");

} catch (Exception ex) {

Logger.getLogger(Window4.class.getName()).log(Level.SEVERE, null, ex);

}

}

else

{

State =true;

//jButton1.setText("Parar toma de datos");

try {

//Modelo.addRow(new Object[]{"1","2","3"});

PanamaHitek\_Arduino.sendData("1");

} catch (Exception ex) {

Logger.getLogger(Window4.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

//Modelo.removeRow(0);

javax.swing.JFileChooser Ventana = new javax.swing.JFileChooser();

String ruta = "";

try {

if (Ventana.showSaveDialog(null)== Ventana.APPROVE\_OPTION)

{

ruta = Ventana.getSelectedFile().getAbsolutePath()+".xls";

FicherosExcel(ruta);

}

} catch (Exception ex)

{

ex.printStackTrace();

}

}

public void FicherosExcel(String input){

HSSFWorkbook libro = new HSSFWorkbook();

HSSFSheet hoja = libro.createSheet();

HSSFRow fila = hoja.createRow(0);

HSSFCell celda = fila.createCell(0);

celda.setCellValue("Datos obtenidos: Paciente");

fila= hoja.createRow(1);

celda = fila.createCell(0);

celda.setCellValue("Ritmo Cardiaco");

celda = fila.createCell(1);

celda.setCellValue("Altura");

celda = fila.createCell(2);

celda.setCellValue("Temperatura");

for(int i=0; i <= Modelo.getRowCount()-1;i++ ){

fila = hoja.createRow(i+2);

for(int j=0;j<=2;j++){

celda = fila.createCell(j);

celda.setCellValue(jTable1.getValueAt(i,j).toString());

}

}

try {

FileOutputStream Fichero = new FileOutputStream(input);

libro.write(Fichero);

Fichero.close();

} catch (Exception e)

{

e.printStackTrace();

}

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable()

{

public void run()

{

new Window4().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

// End of variables declaration

}

Interfaz

