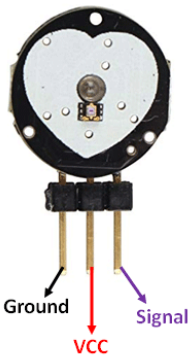
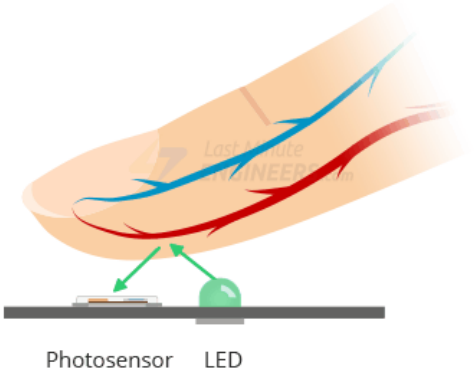
Arduino a senzor pulzu

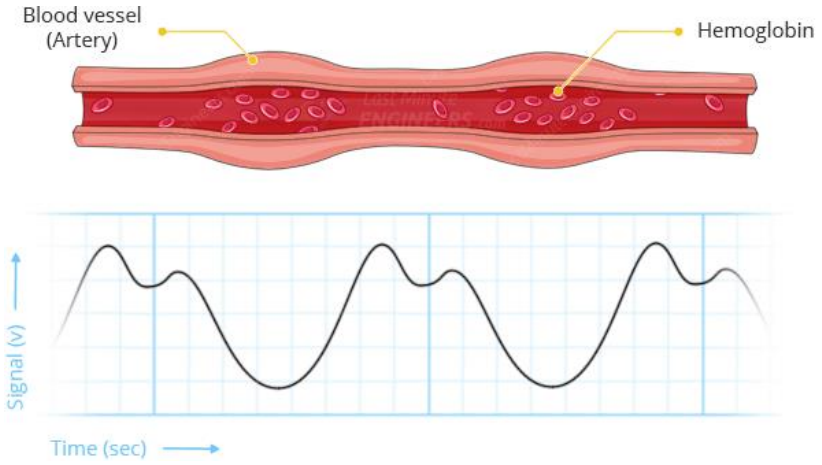
1. Princíp činnosti

Senzor pulzu (pulse sensor) je nízkoenergetický snímač srdcovej frekvencie typu plug-and-play pre Arduino. Na prednej strane je otvor, v ktoro svieti zelená LED dióda. Pod ňou je snímač svetla, fotosenzor. Senzor pulzu priložíme na prst - na prst svieti zelené svetlo (~ 550 nm), ktoré sa odráža a množstvo odrazeného svetla sa meria pomocou fotosenzora.

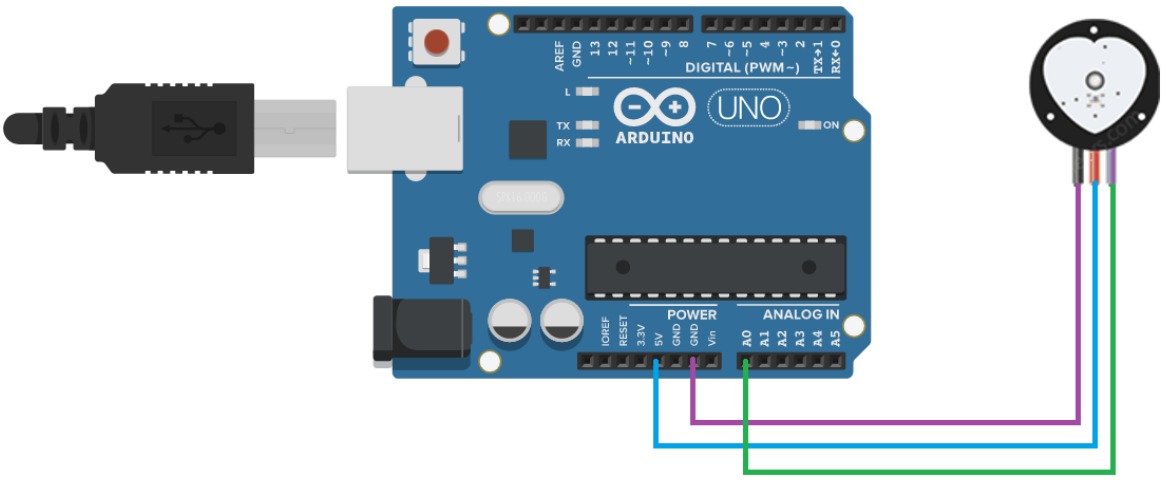




Okysličená krv je červenšia ako neokysličená a preto lepšie absorbuje zelené svetlo. Ako je krv pumpovaná cez prst pri každom údere, mení sa množstvo odrazeného svetla a od toho závisí hodnota výstupu senzora. Táto metóda detekcie impulzov svetlom sa nazýva fotopletyzmogram.

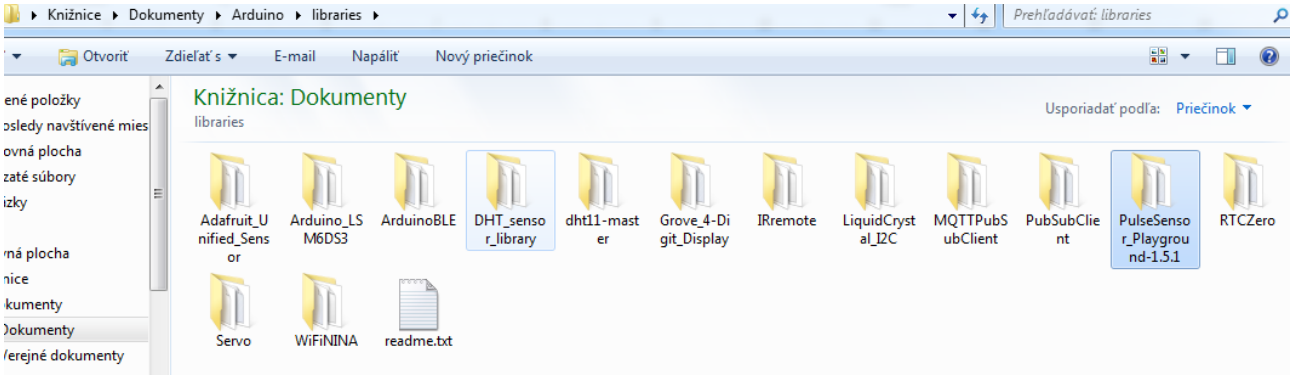


2. Schéma zapojenia

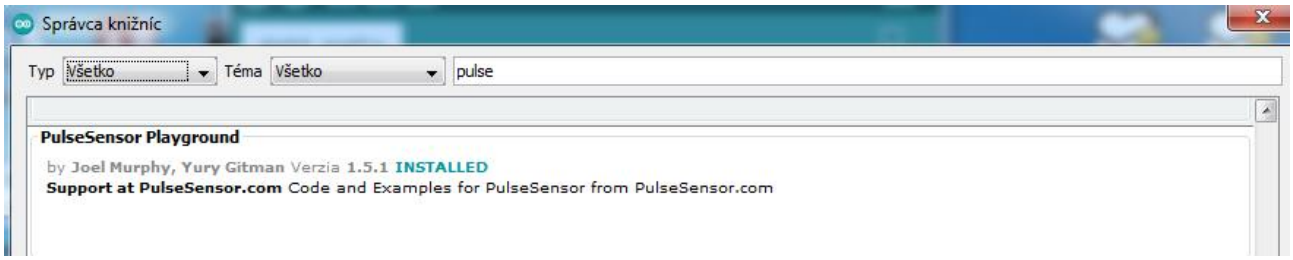


3. Inštalácia knižnice

Knižnicu **PulseSenzor.h** stiahneme odtiaľto https://www.arduinolibraries.info/libraries/pulse-sensor-playground ako zip súbor. Rozbalíme je do priečinka ***Libraries***:



V správcovi knižníc si overíme, že sa doinštalovala:



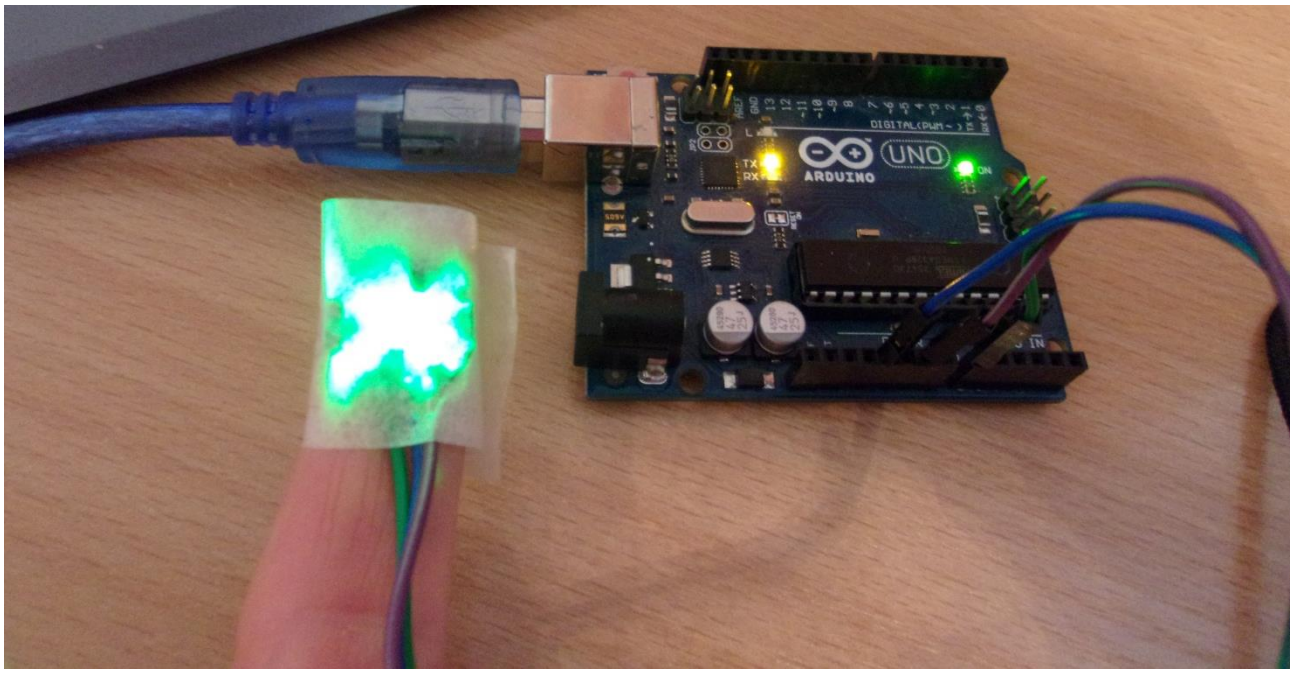
V menu **Súbor-Príklady** nájdeme vzorové programy, ktoré môžeme vyskúšať a obmieňať.

4. Program na nastavenie prahovej hodnoty

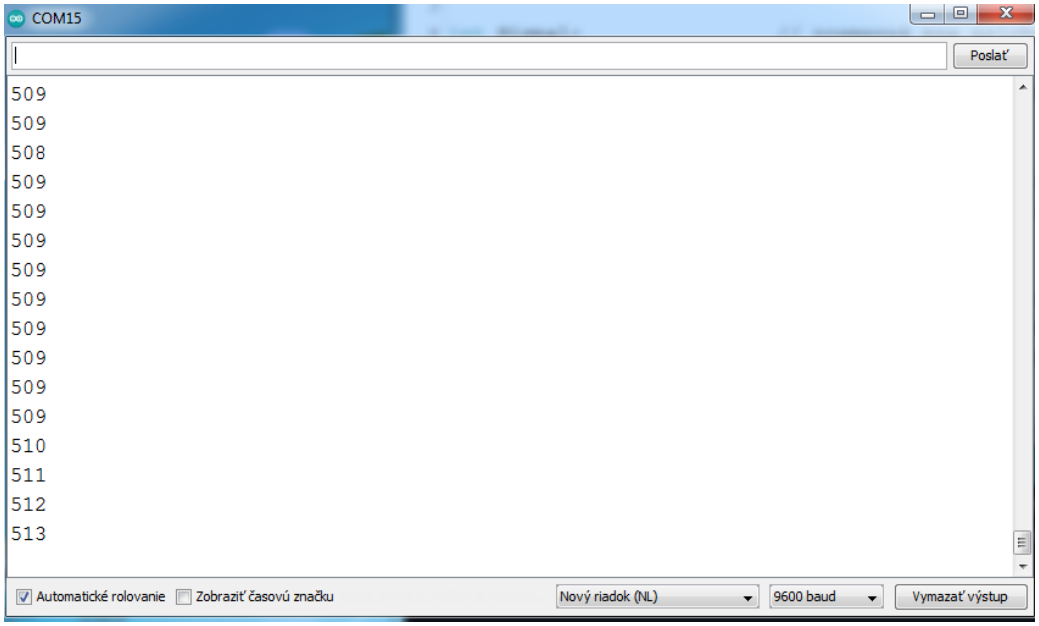
V tomto programe bliká interná LED dióda v rytme srdcového pulzu.

```
int PulseSensor = A0;      // na A0 je pripojený vodič so signálom zo senzora pulzu
int LED13 = 13;            //  interná LED na Arduino doske

int Signal;                // premenná pre prichádzajúce dáta v rozsahu 0-1024
int Threshold = 511;       //prahová hodnota, ktorá určuje, čo sa bude počítať ako úder a
                           // čo sa bude ignorovať

void setup() {
  pinMode(LED13,OUTPUT);      // nastavenie výstupu na internú LED
  Serial.begin(9600);         // nastavenie rýchlosti sériovej komunikácie
}
void loop() {
  Signal = analogRead(PulseSensor); // prečítanie hodnoty zo senzora
  Serial.println(Signal);           // výpis hodnoty
  if(Signal > Threshold){           // interná LED bude blikať v rytme srdcového pulzu
     digitalWrite(LED13,HIGH);
   } else {
     digitalWrite(LED13,LOW);               //
   }
delay(10);
}
```

Snímač prilepíme jemne na prst.

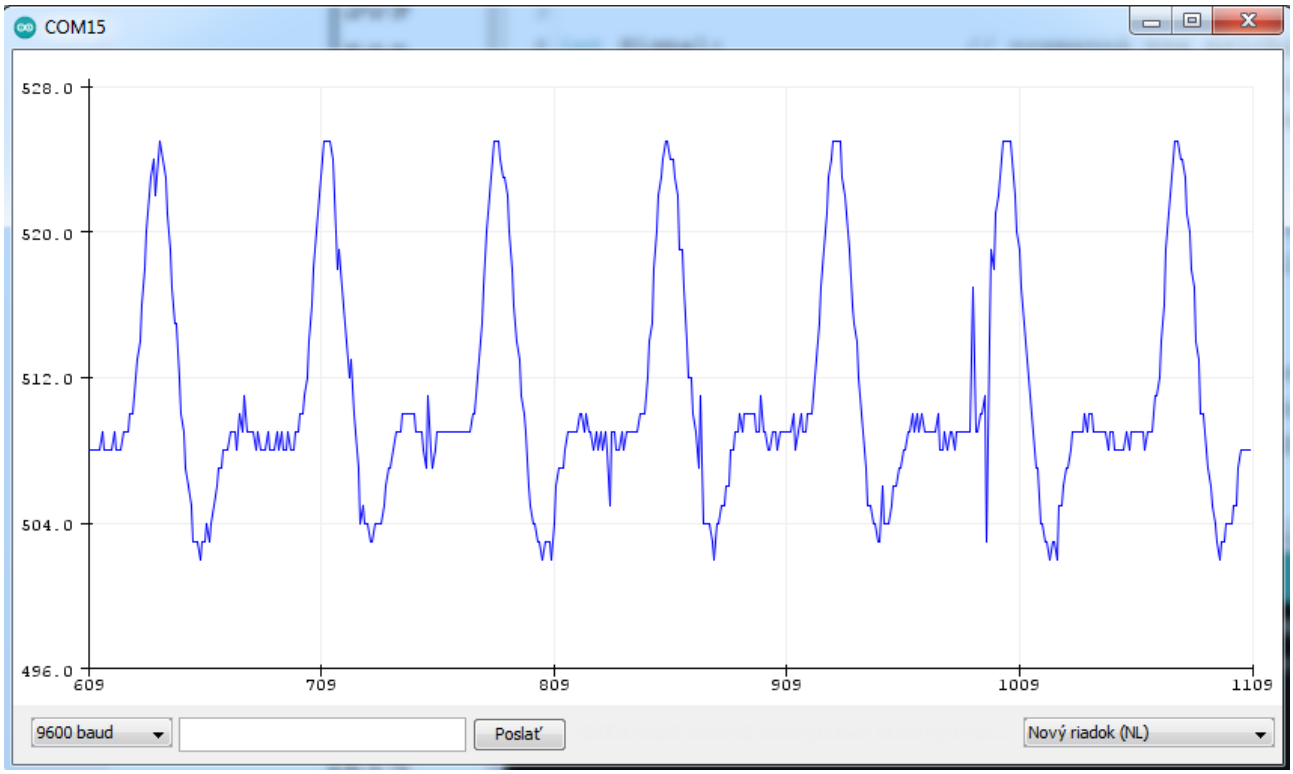


V programe treba experimentovať s prahovou hodnotou *Threshold* podľa výsledkov, ktoré vidíme v sériovom monitore. Na sériovom monitore si pozrieme namerané hodnoty a premennú Threshold nastavíme na prostrednú hodnotu zo všetkých nameraných.



Ak už máme zistenú a nastavenú hodnotu premennej Threshold, program znovu skompilujeme a nahráme do Arduina.

Namiesto sériového monitora si zapneme sériový plotter:



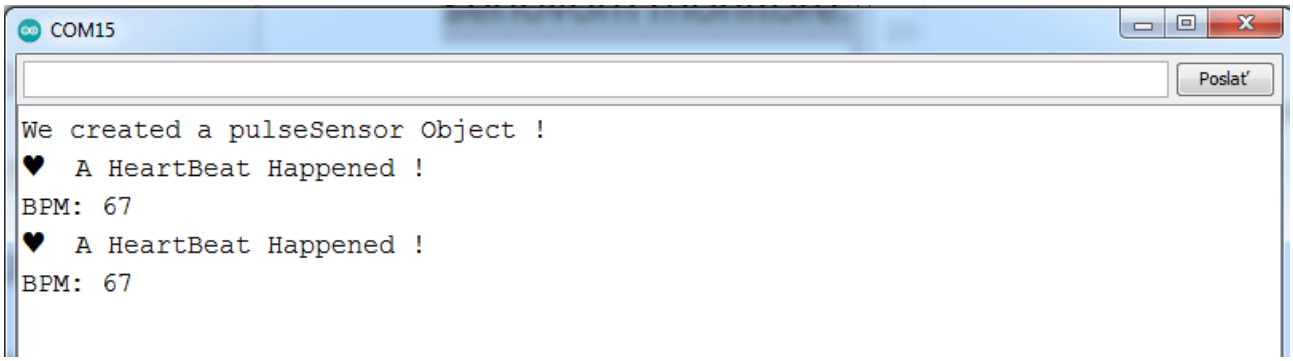
5. Program Getting BPM to monitor

BPM = *beats per minute* = *počet úderov za minútu*. Tento program vypočítava čas medzi impulzmi na získanie srdcovej frekvencie a zobrazuje ju na sériovom monitore. Hodnotu premennej *Threshold* treba nastaviť na základe skúsenosti z predchádzajúceho programu.

```
/*  Getting_BPM_to_Monitor prints the BPM to the Serial Monitor, using the least lines of code
and PulseSensor Library.
 *  Tutorial Webpage: https://pulsesensor.com/pages/getting-advanced
 *
--------Use This Sketch To-----------------------------------------
1) Displays user's live and changing BPM, Beats Per Minute, in Arduino's native Serial Monitor.
2) Print: "♥  A HeartBeat Happened !" when a beat is detected, live.
2) Learn about using a PulseSensor Library "Object".
4) Blinks LED on PIN 13 with user's Heartbeat.
-----------------------------------------------------------------------*/

#define USE_ARDUINO_INTERRUPTS true    // Set-up low-level interrupts for most acurate BPM math.
#include <PulseSensorPlayground.h>     // Includes the PulseSensorPlayground Library.

//  Variables
const int PulseWire = 0;       // PulseSensor PURPLE WIRE connected to ANALOG PIN 0
const int LED13 = 13;          // The on-board Arduino LED, close to PIN 13.
int Threshold = 512;           // Determine which Signal to "count as a beat" and which to
ignore.

PulseSensorPlayground pulseSensor;  // Creates an instance of the PulseSensorPlayground object
called "pulseSensor"

void setup() {
  Serial.begin(9600);          // For Serial Monitor
  pulseSensor.analogInput(PulseWire);
  pulseSensor.blinkOnPulse(LED13);  //blink Arduino's LED with heartbeat.
  pulseSensor.setThreshold(Threshold);
```

```
  // Double-check the "pulseSensor" object was created and "began" seeing a signal.
   if (pulseSensor.begin()) {
    Serial.println("We created a pulseSensor Object !");  //This prints one time at Arduino
power-up,  or on Arduino reset.
  }
}

void loop() {
 int myBPM = pulseSensor.getBeatsPerMinute();  // Calls function on our pulseSensor object that
returns BPM as an "int".
                                               // "myBPM" hold this BPM value now.
 if (pulseSensor.sawStartOfBeat()) {            // Constantly test to see if "a beat happened".
   //Serial.println("♥  A HeartBeat Happened ! "); // If test is "true", print a message "a
heartbeat happened".
   //Serial.print("BPM: ");                        // Print phrase "BPM: "
   Serial.println(myBPM);                        // Print the value inside of myBPM.
  }
  delay(20);                    // considered best practice in a simple sketch.
}
```

Po nahratí programu do Arduina si zapneme sériový monitor:



Ak chceme hodnoty odoslané z Arduina spracovávať v Node-RED, výstup zjednodušíme takto:

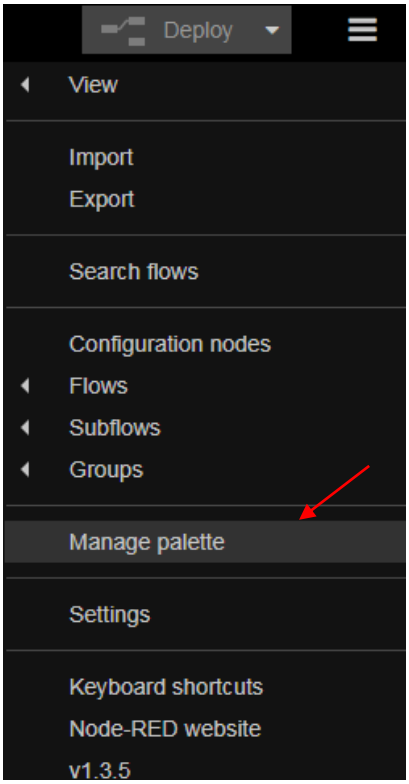
```
35  if (pulseSensor.sawStartOfBeat()) {            // Co
36    //Serial.println("♥  A HeartBeat Happened ! "); //
37    //Serial.print("BPM: ");                        //
38    Serial.println(myBPM);                       // P
39   }
40   delay(20);                    // considered best pra
41 }
```

Arduino IDE môžeme zavrieť a spustiť si Node-RED.

6. Node-RED

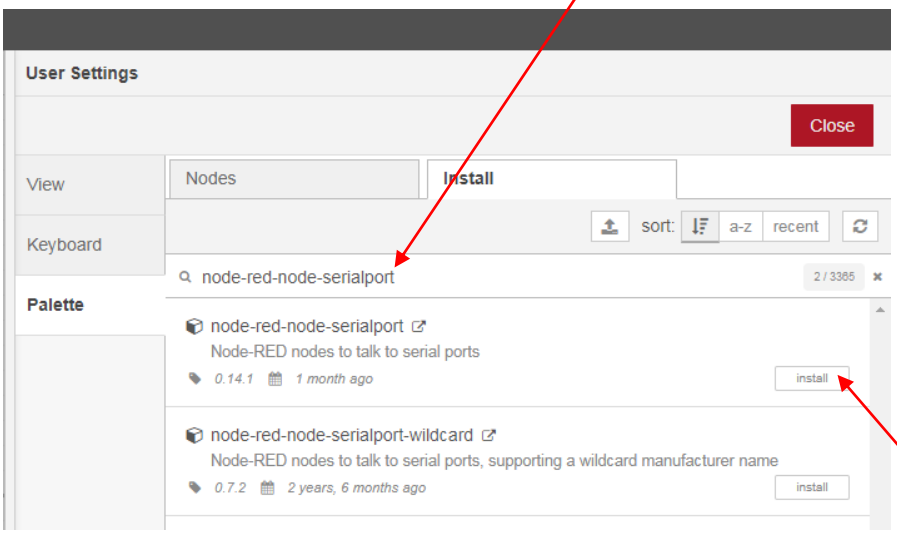
V Príkazovom riadku zadáme príkaz `node-red`

Otvoríme si nejaký internetový prehliadač a zadáme adresu `localhost:1880`



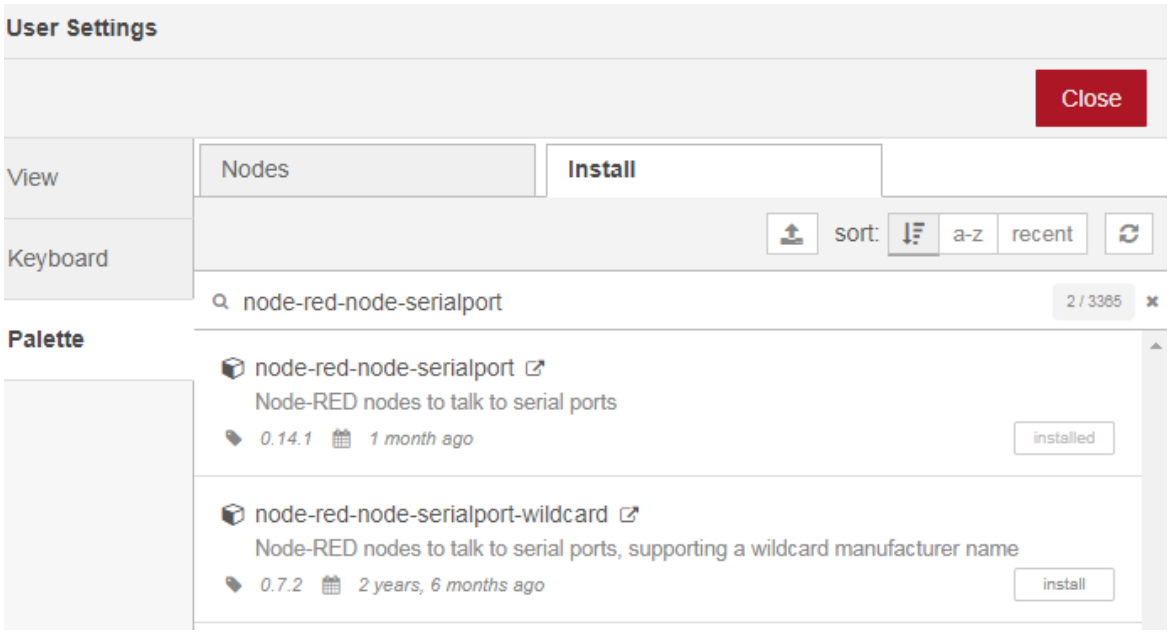
Node-RED nemá predinštalované uzly pre Arduino a Serial Port. Doinštalujeme ich pomocou Manažéra paliet:

Sem napíšeme iba slovo *serialport*, nájde nám 2 palety.

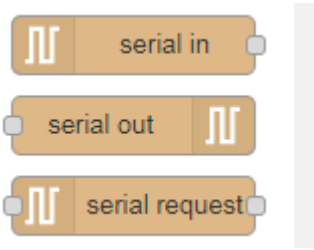


Túto nainštalujeme.

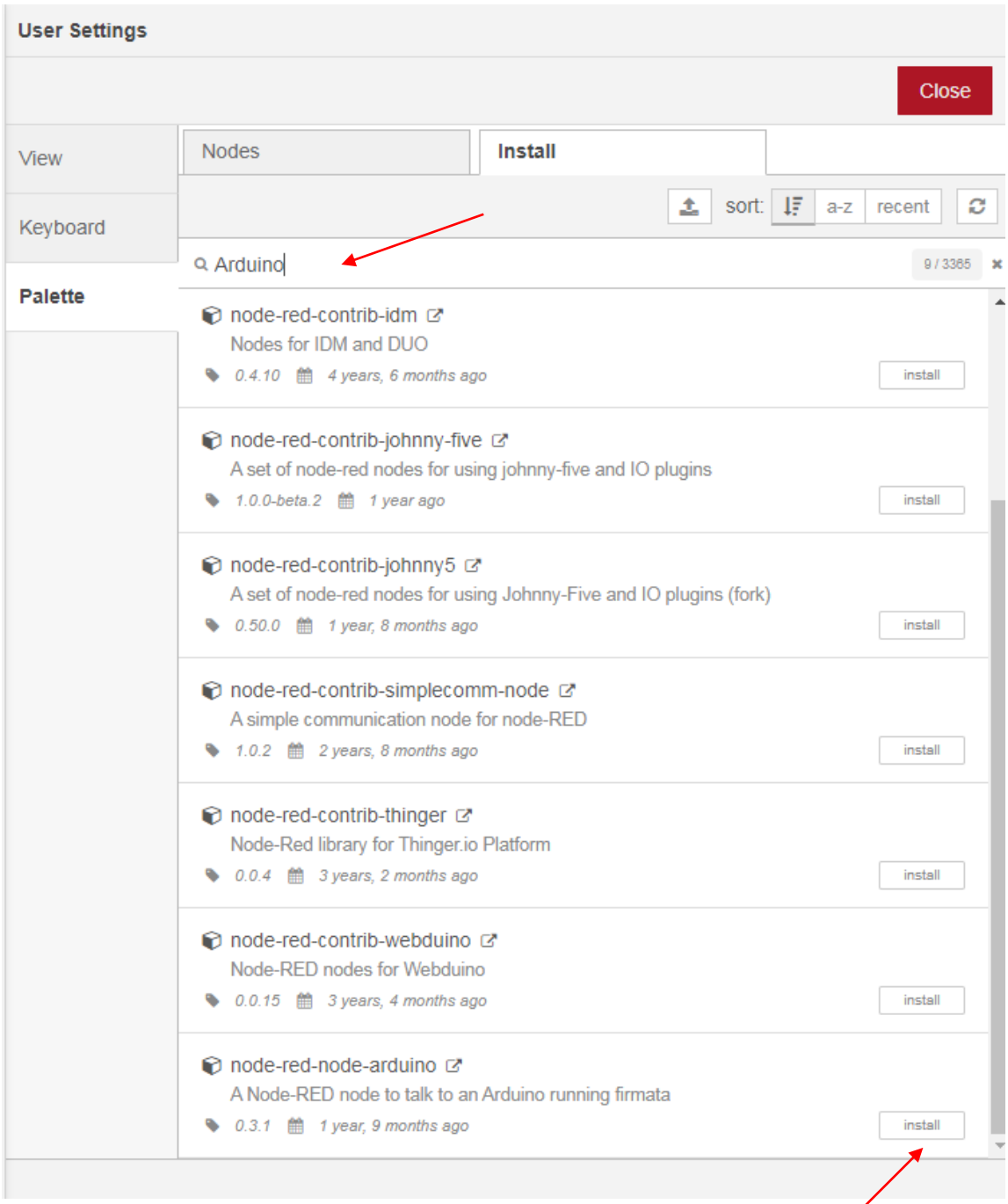
Po nainštalovaní to vyzerá takto:



V paneli uzlov, v skupine network nám pribudnú tieto uzly:

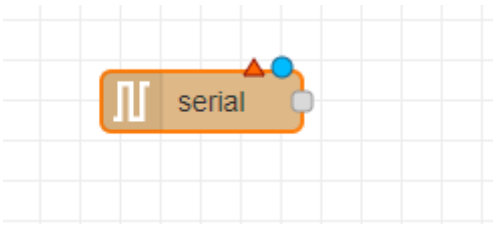


Podobne doinštalujeme uzly pre Arduino:

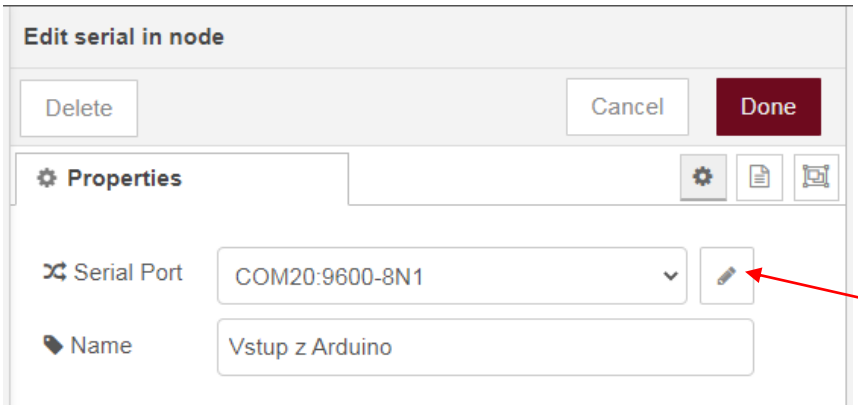


Túto nainštalujeme.

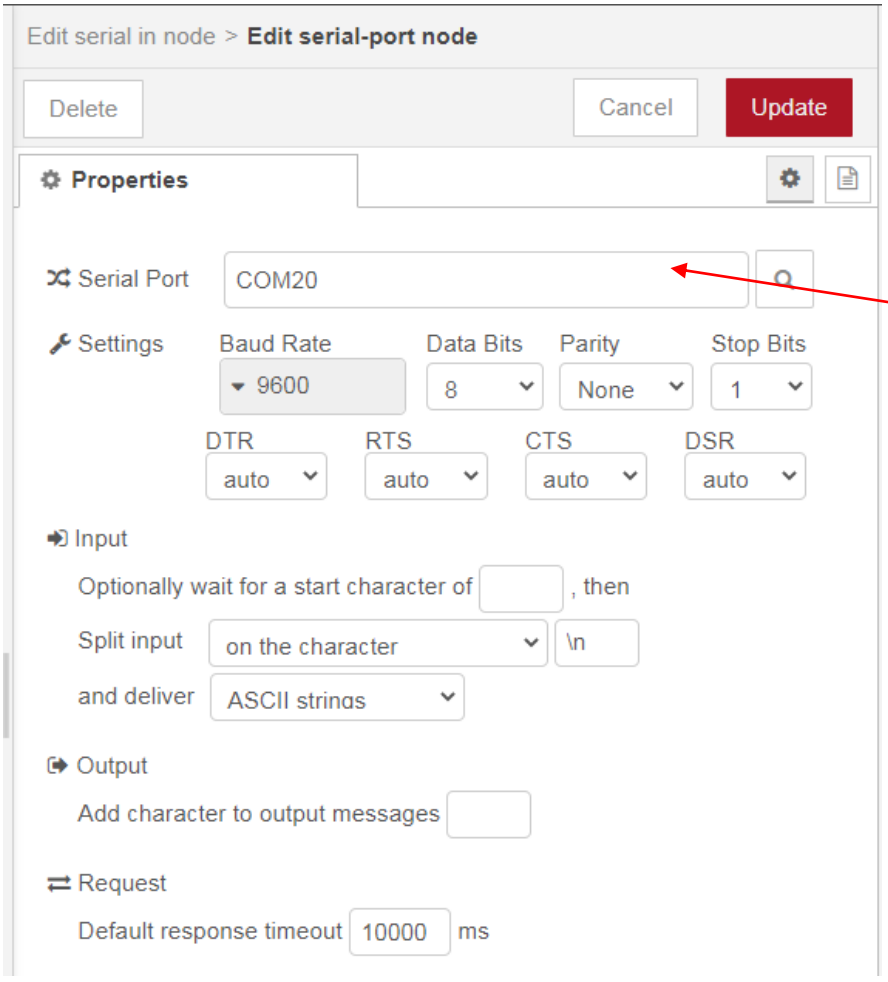
Na pracovnú plochu vložíme uzol **serial in**:



Zadáme mu takéto vlastnosti:

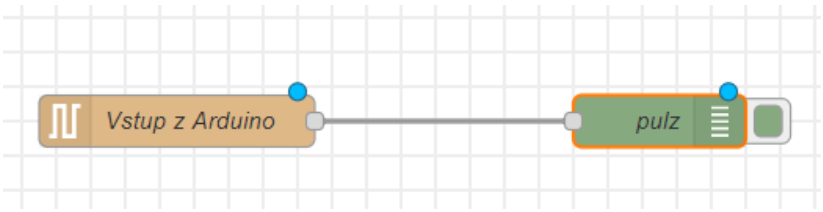


Klikneme na ikonu ceruzky.

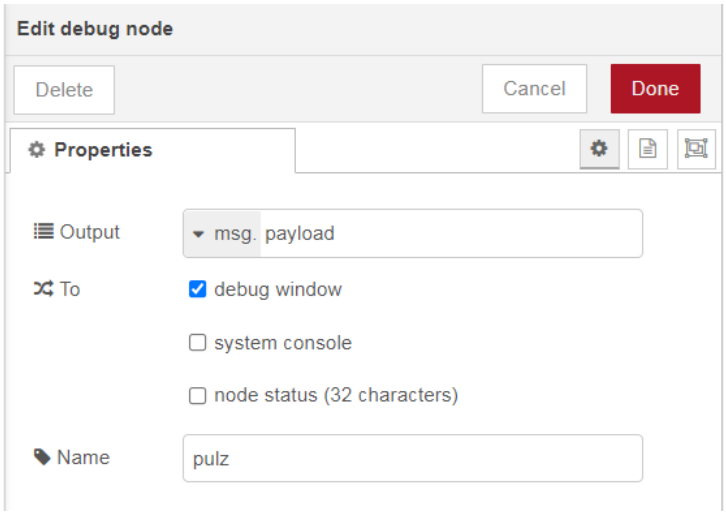


Zadáme taký port, na ktorom máme pripojené Arduino..

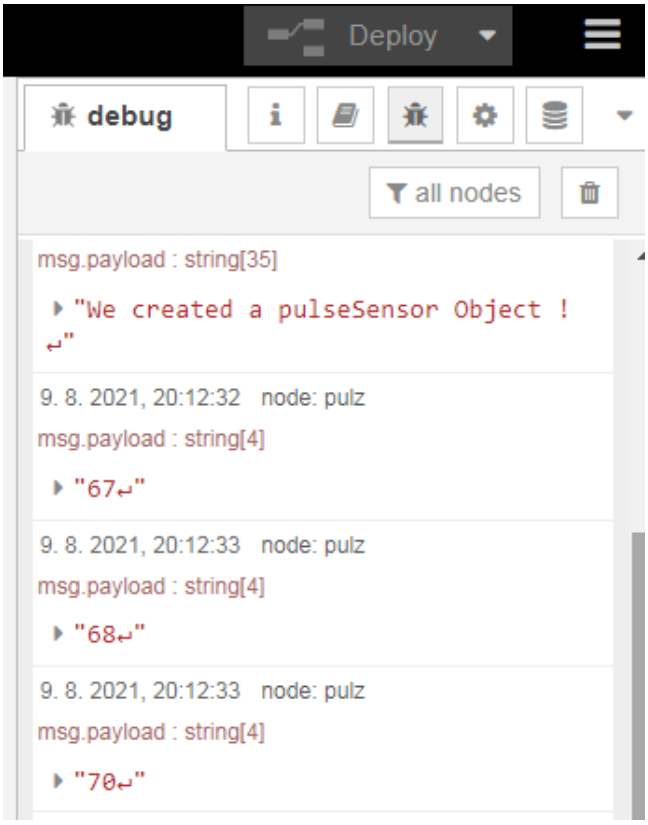
Pridáme uzol **debug**:



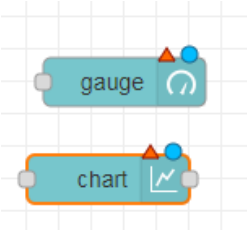
Vlastnosti uzla **debug**:



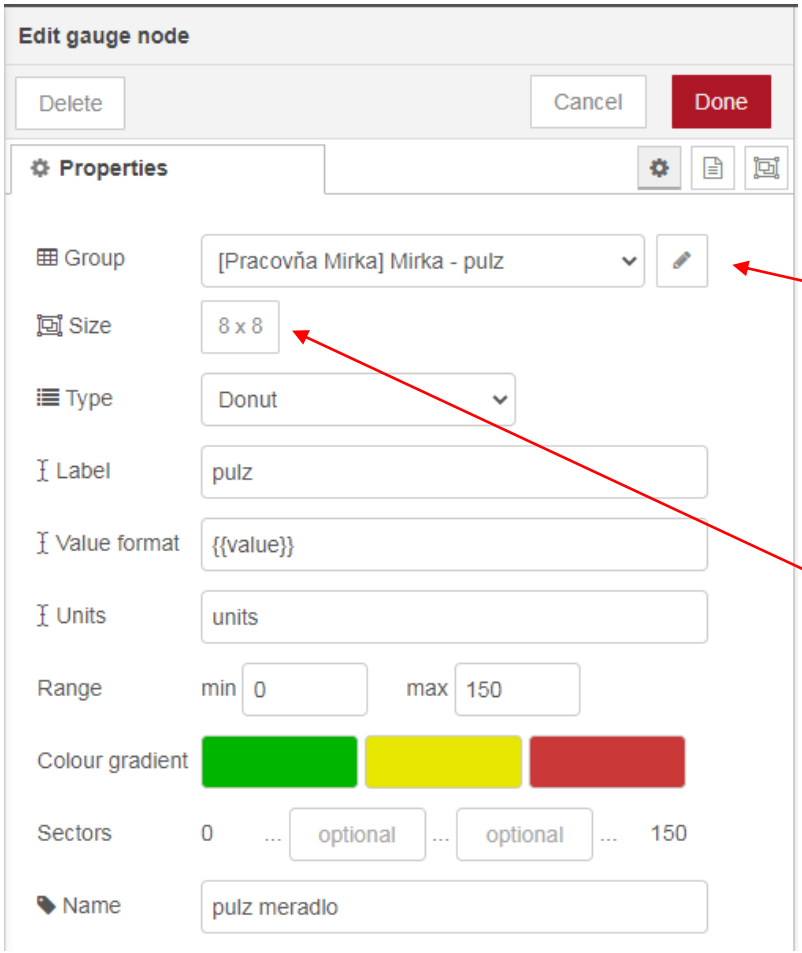
Po kliknutí na **Deploy** sa v debug okne každú 1 sekundu zobrazujú hodnoty pulzu zo senzoru, pripojenom k Arduinu (pozor, Arduino sme naprogramovali tak, že vysiela hodnoty každých 20 milisekúnd):



Pridáme uzly **gauge** a **chart**:



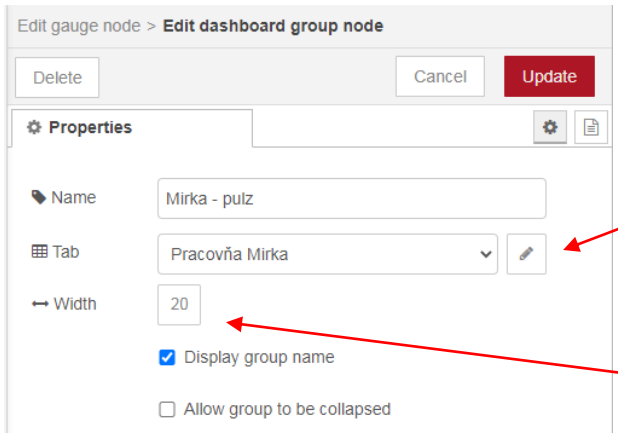
Vlastnosti uzla **gauge** (ciferník):



Klikneme na ikonu ceruzky. Dostaneme sa do ďalšieho dialógového okna, kde vyberáme tabuľku (jej názov je v tej hranatej zátvorke) a zadávame meno skupiny (to je uvedené za zátvorkou.

Nezabudneme nastaviť veľkosť a typ ciferníka.

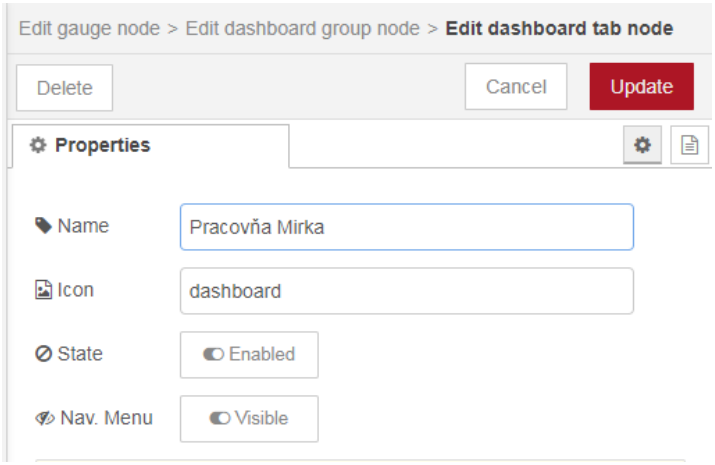
Zadanie mena skupiny (group node):



Klikneme na ikonu ceruzky. Dostaneme sa do ďalšieho dialógového okna, kde vyberáme tabuľku. Ak tabuľku ešte nemáme vytvorenú, tak ju nemáme z čoho vybrať, takže ju musíme vytvoriť.

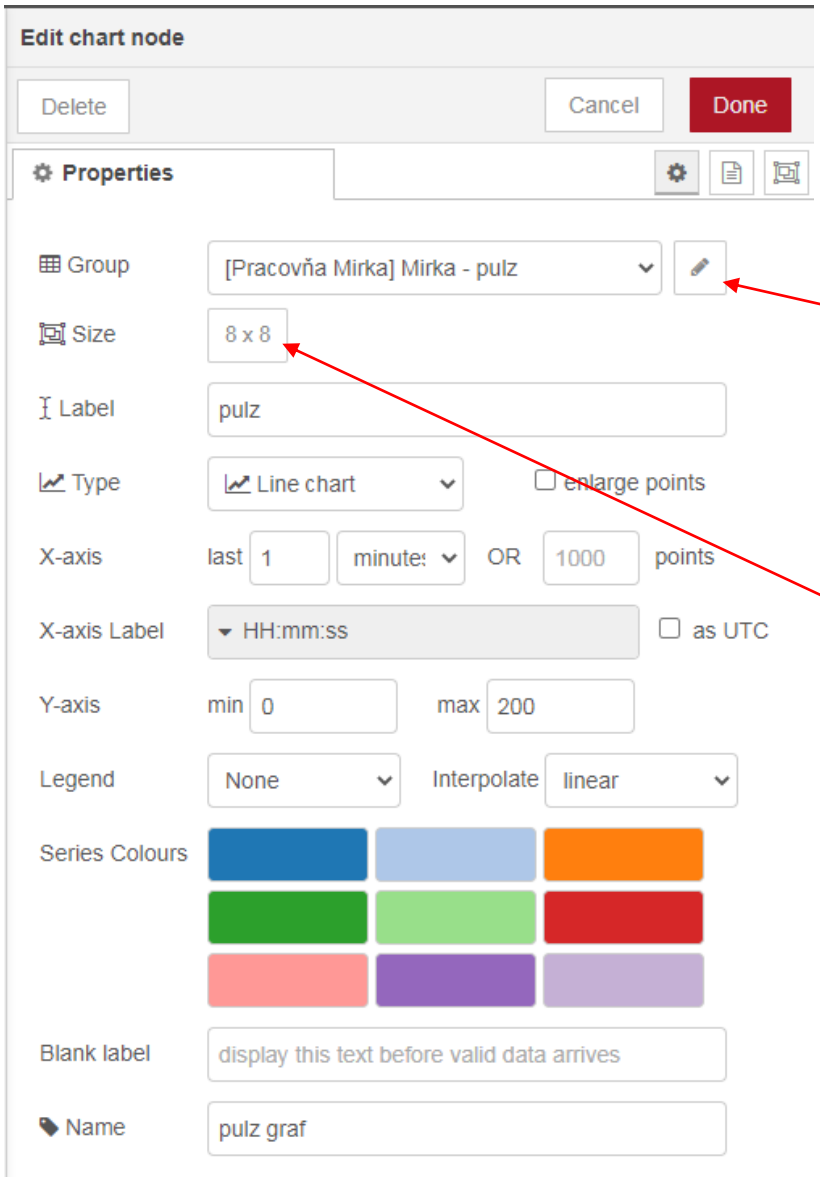
Nezabudneme nastaviť šírku tabuľky.

Výber alebo vytvorenie tabuľky (tab node):



Z dialógových panelov sa vraciame do vyššej úrovne cez tlačidlo **Update** resp. **Done**.

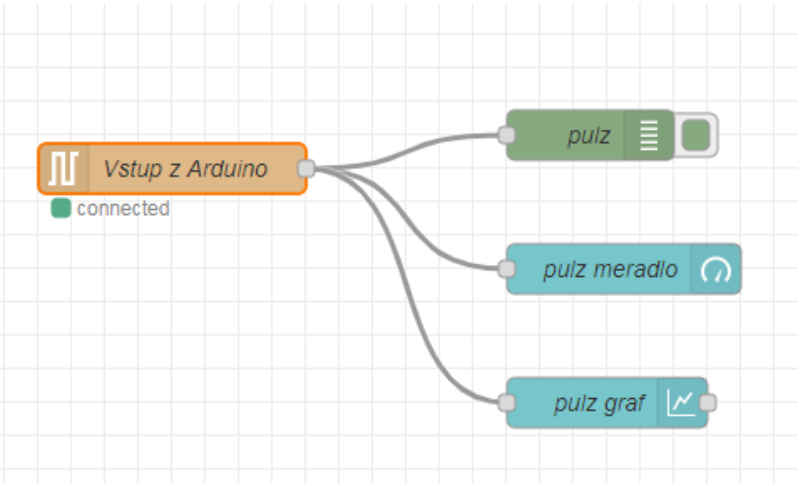
Vlastnosti uzla **chart** (graf):



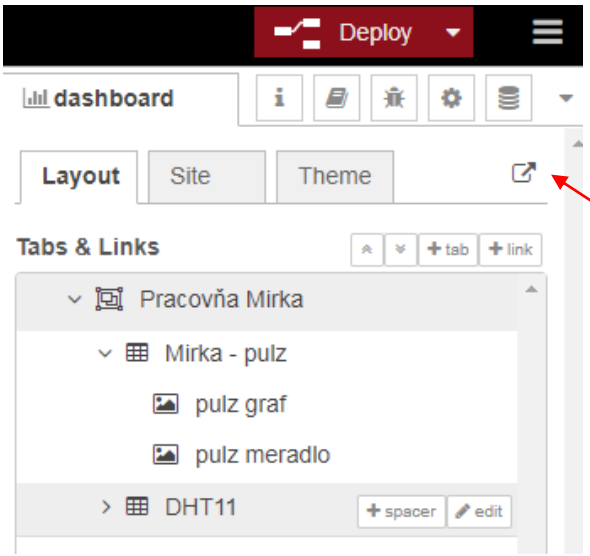
Klikneme na ikonu ceruzky. Dostaneme sa do ďalšieho dialógového okna, kde vyberáme tabuľku (jej názov je v tej hranatej zátvorke) a zadávame meno skupiny (to je uvedené za zátvorkou.

Nezabudneme nastaviť veľkosť a typ grafu.

Takže máme vytvorený takýto **flow** (tok):

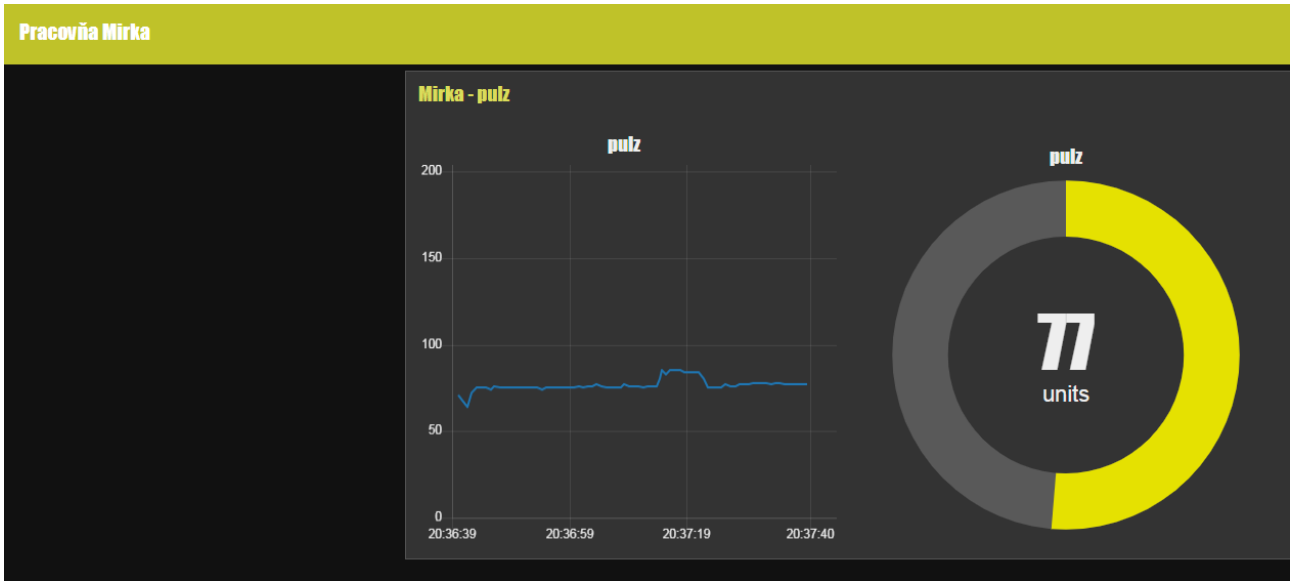


Po kliknutí na **Deploy** si zobrazíme **Dashboard** (palubnú dosku):



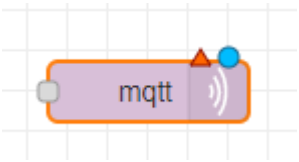
Dashboard sa zobrazí v novom okne prehliadača.

Výsledok:

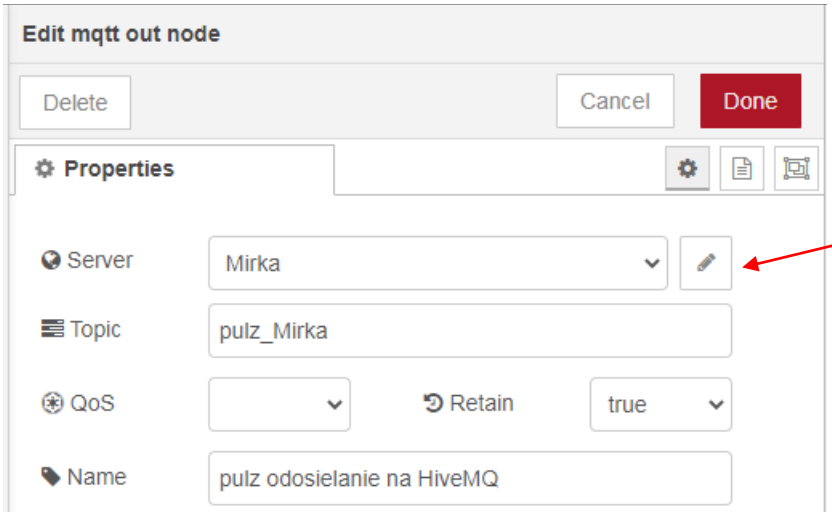


7. Publikovanie na HiveMQ

Pridáme uzol **mqtt out**:

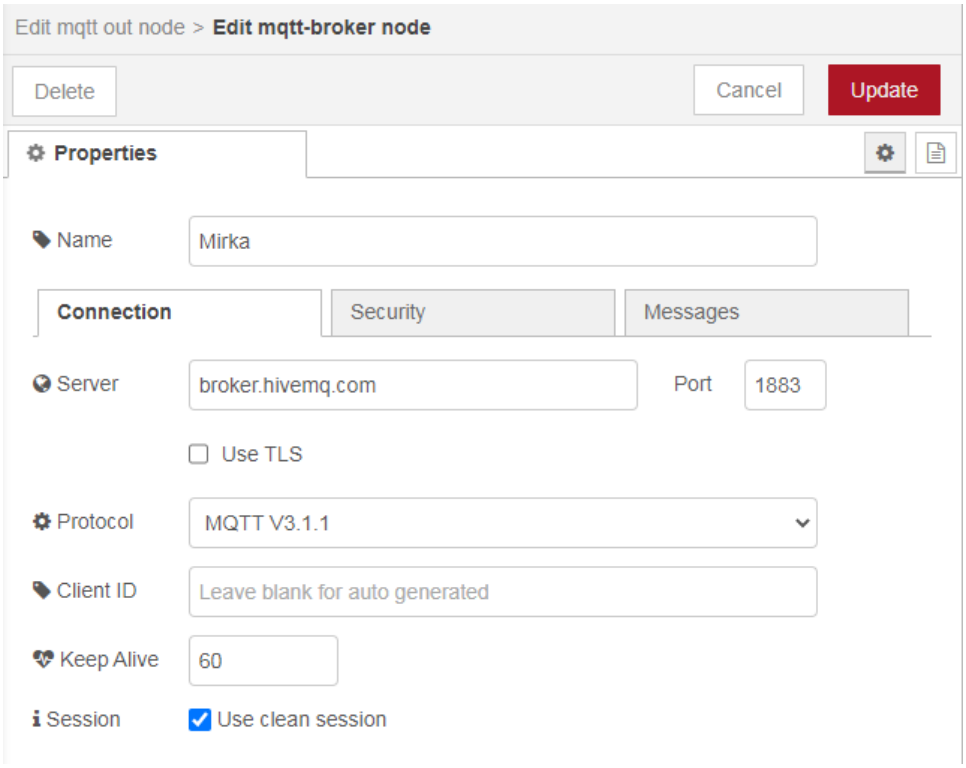


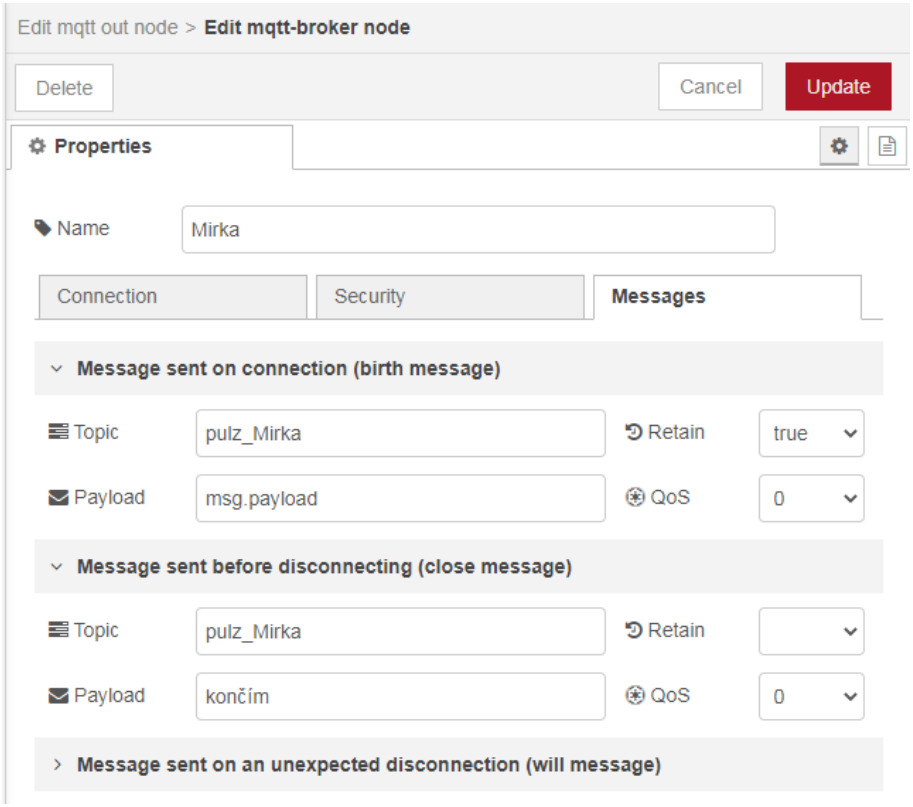
Vlastnosti uzla **mqtt out**:



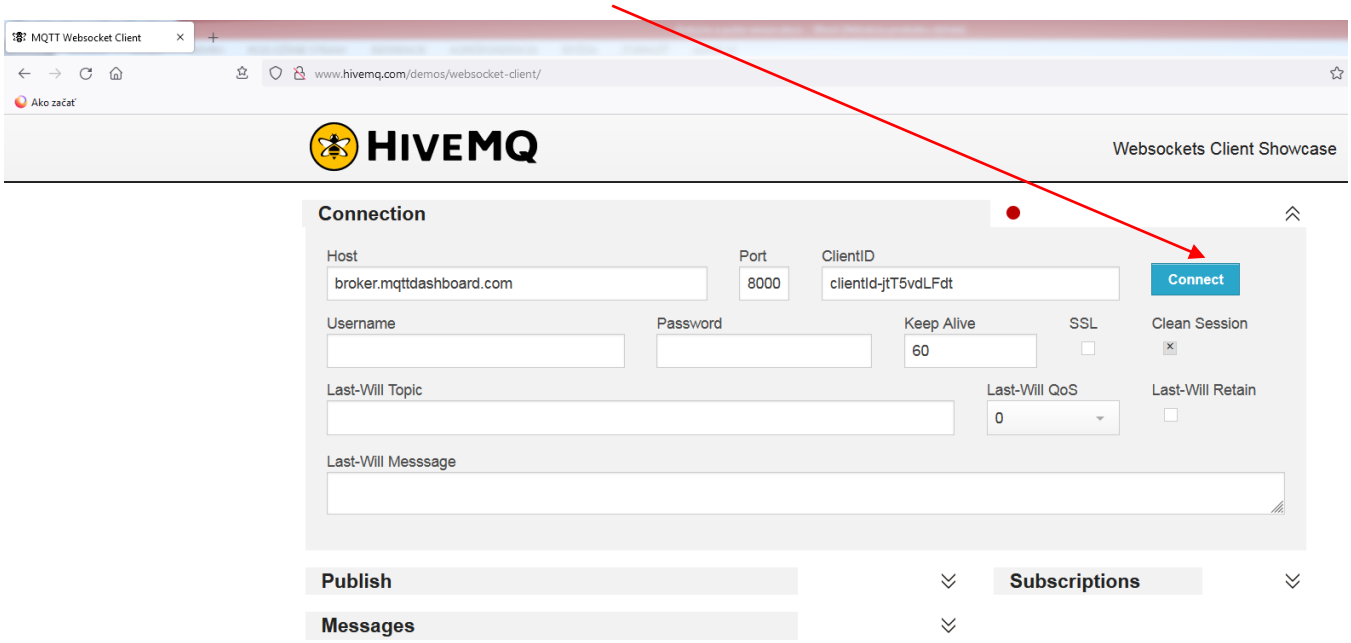
Meno servera zadáme v ďalšom dialógovom okne, ktoré sa zobrazí po kliknutí na ikonu ceruzky.

Nastavenie mqtt brokera:

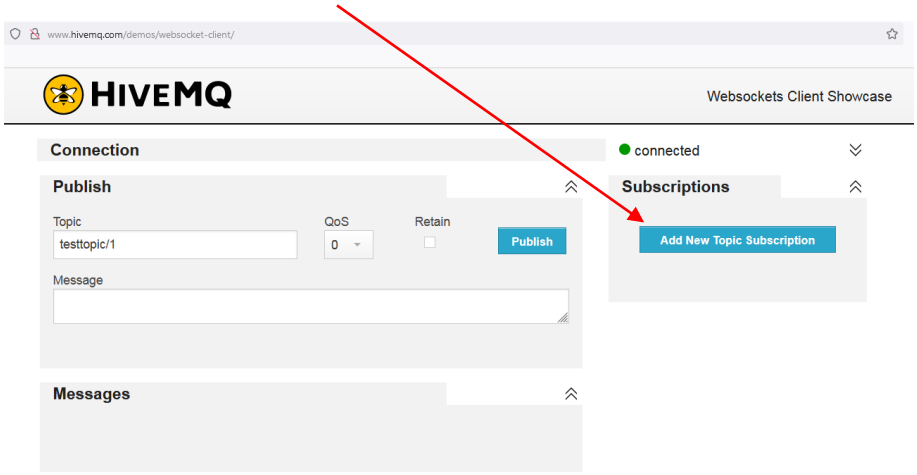




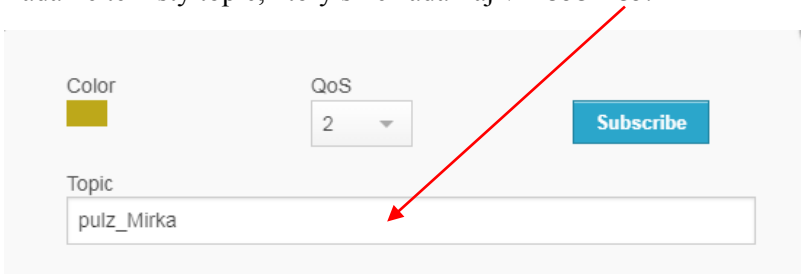
Pripojíme sa na brokera HiveMQ http://www.hivemq.com/demos/websocket-client/ a klikneme na tlačidlo **Connect**:



Po pripojení zadáme odber témy (topic):



Zadáme ten istý topic, ktorý sme zadali aj v Node-Red:



V okne Messages sa budú zobrazovať hodnoty, ktoré Node-Red vslal po prijatí z Arduina: