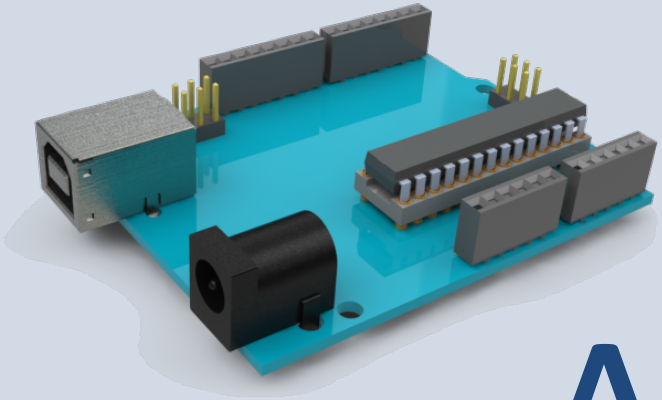
https://www.halvorsen.blog



Arduino

Hans-Petter Halvorsen

Table of Contents

Introduction to Arduino

Hans-Petter Halvorsen

Table of Contents

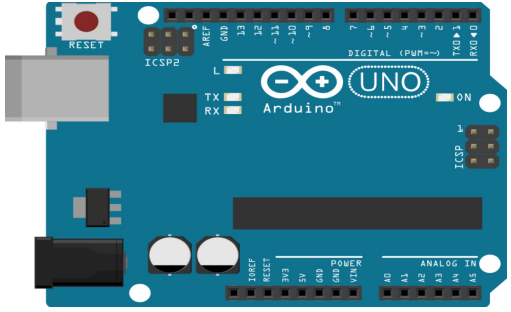
Arduino

- Arduino is an open-source electronics platform based on easy-to-use hardware and software.
- It's intended for anyone making interactive projects, from kids to grown-ups.
- You can connect different Sensors, like Temperature, etc.
- It is used a lots in Internet of Things projects
- Homepage: https://www.arduino.cc

Arduino

- Arduino is a Microcontroller
- Arduino is an open-source platform with Input/Output Pins (Digital In/Out, Analog In and PWM)
- Price about $20
- Arduino Starter Kit ~$40-80

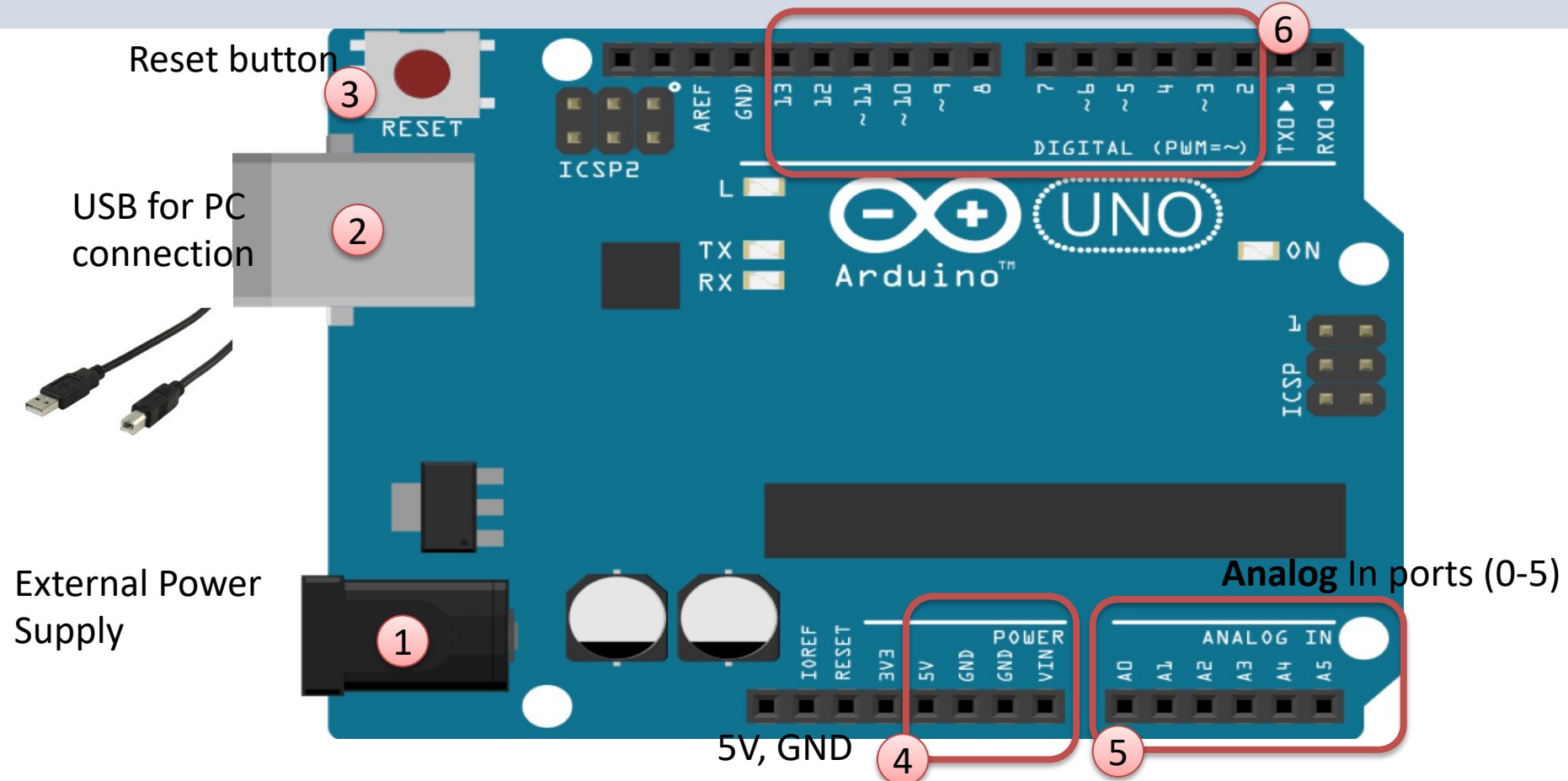
  with Cables, Wires, Resistors, Sensors, etc.



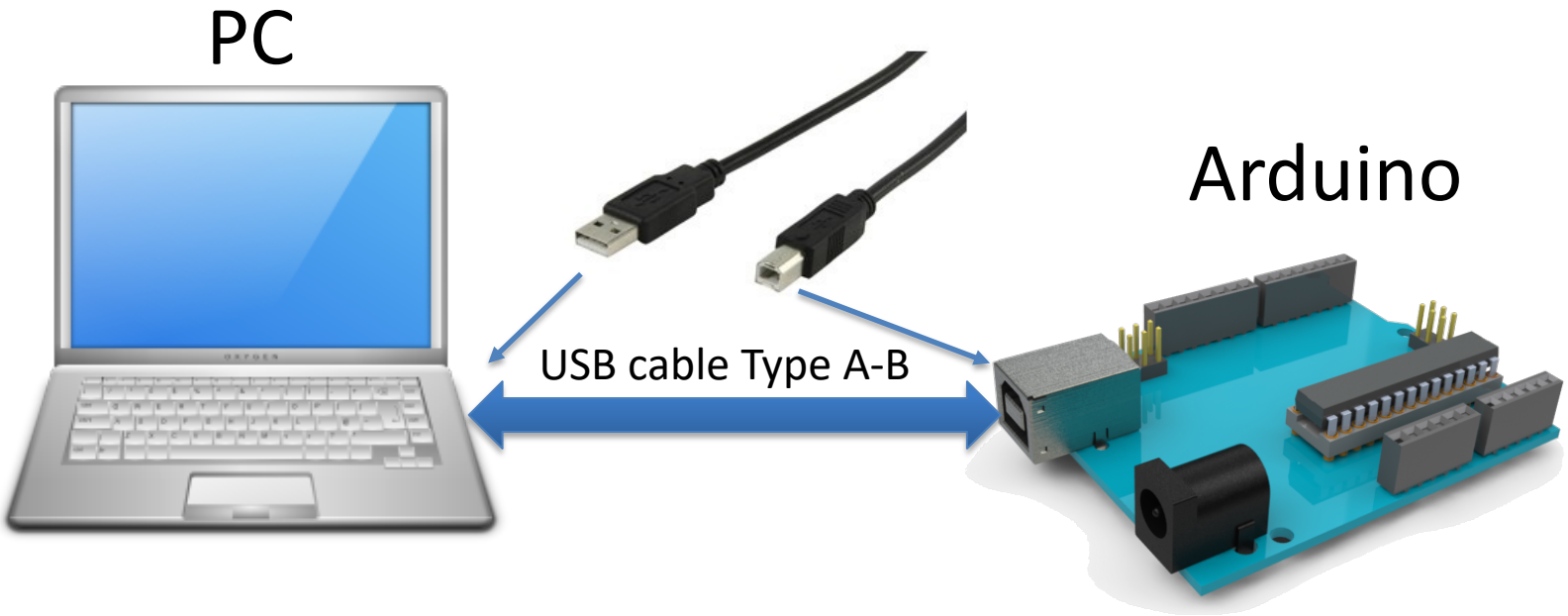
Arduino

- Lots of different Arduino boards exists
- There are different Arduino boards with different features and boards that are tailormade for different applications
- https://www.arduino.cc/en/Main/Products
- The most common is called “Arduino UNO”

Arduino UNO


Reset button
3
RESET
USB for PC connection
2
External Power Supply
1
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
6
L
TX
RX
Arduino™
UNO
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
5V, GND
4
ANALOG IN
A0
A1
A2
A3
A4
A5
5
Digital ports (2-13)
Analog In ports (0-5)

Connect Arduino to your PC



Arduino Software

Upload Code to Arduino Board

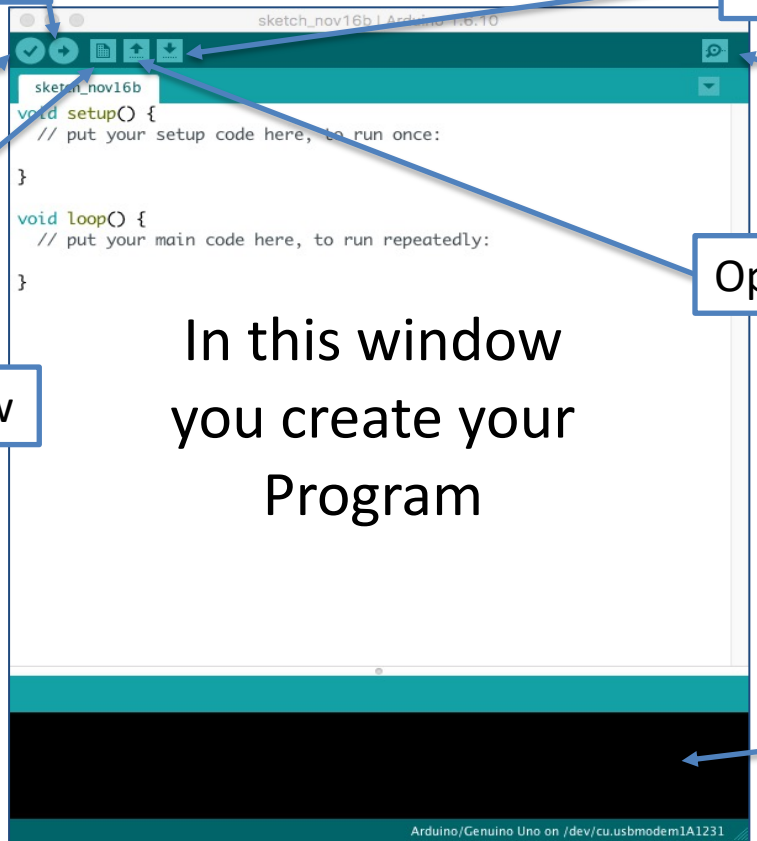
Save

Open Serial Monitor

Compile and Check if Code is OK

Open existing Code

Creates a New Code Window



Error Messages can be seen here

The software can be downloaded for free:

www.arduino.cc

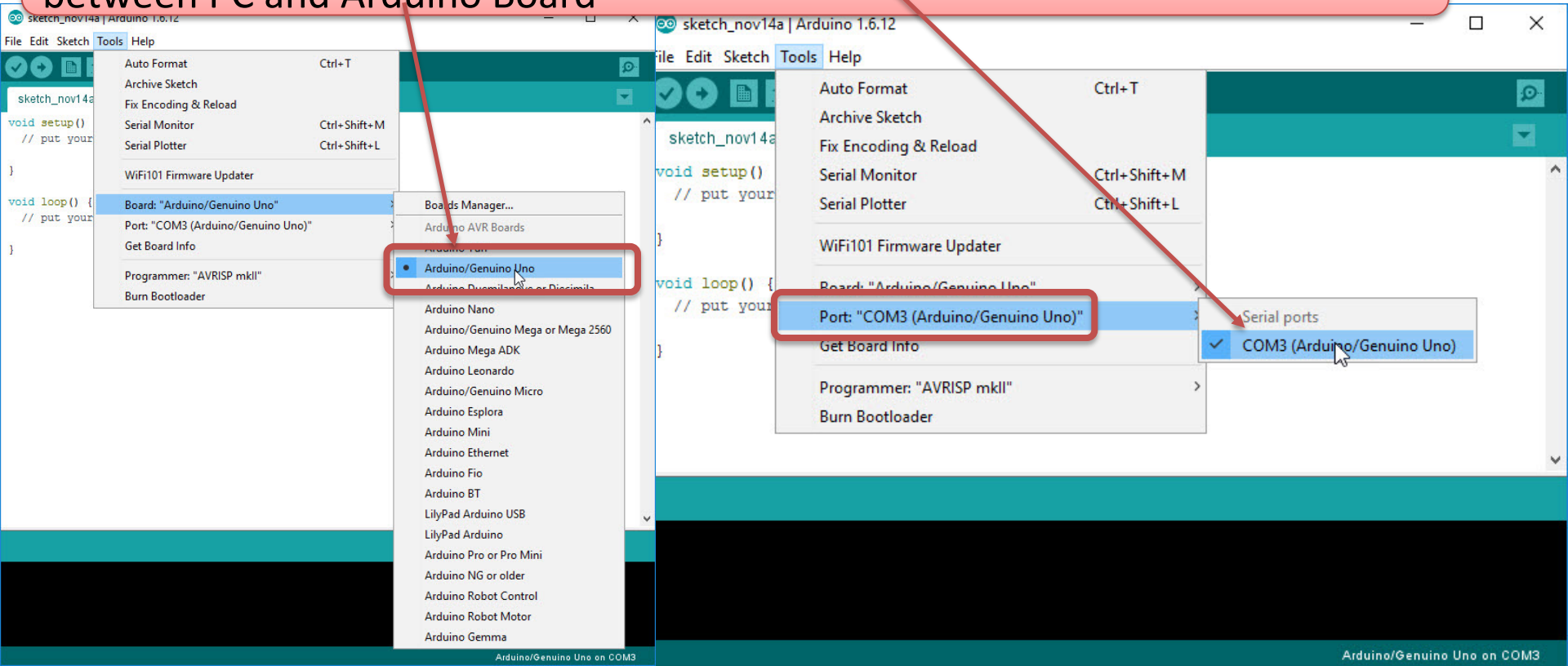
Arduino Programs

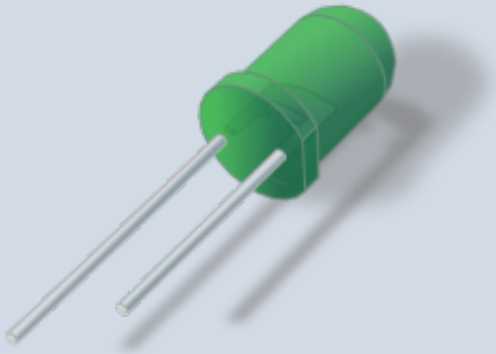
All Arduino programs must follow the following main structure:

```
// Initialization, define variables, etc.

void setup()
{
      // Initialization
      ...
}

void loop()
{
      //Main Program
      ...
}
```

Do you get an Error?

Choose correct Board (Arduino UNO) and the correct Port for Communication between PC and Arduino Board



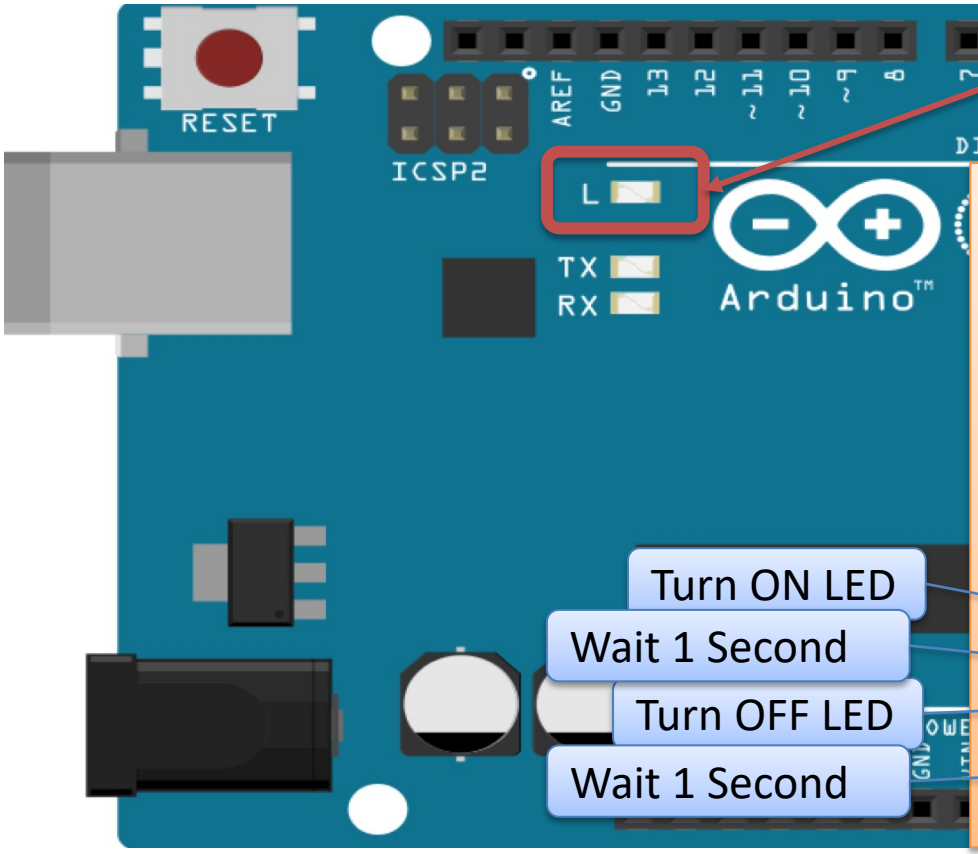


LED

Hans-Petter Halvorsen

Table of Contents

Blinking LED Example



Arduino UNO has a built-in LED that we can control

```
int ledPin = LED_BUILTIN;
int timerWait = 100;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);
  delay(timerWait);
  digitalWrite(ledPin, LOW);
  delay(timerWait);
}
```

Turn ON LED

Wait 1 Second

Turn OFF LED

Wait 1 Second

Code Comments

```
void setup()
{
        pinMode(13, OUTPUT);
}

void loop()
{
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
}
```

Which Pin (0, 1, 3, ...) are you using?

```
pinMode(pin, mode);
```

A Digital Pin can either be an INPUT or an OUTPUT. Since we shall use it to turn-on a LED, we set it to OUTPUT.

```
digitalWrite(pin, value);
```

Turn-on LED

Turn-off LED

A Digital Pin can have 2 values, either **HIGH** or **LOW**

```
delay(ms);
```

The delay() function makes a small pause in milliseconds (ms), e.g., delay(1000) pause the program for 1 second

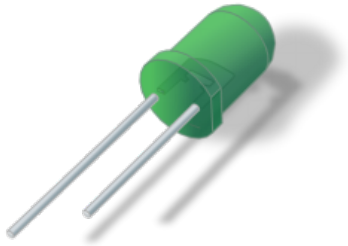
External LED Example

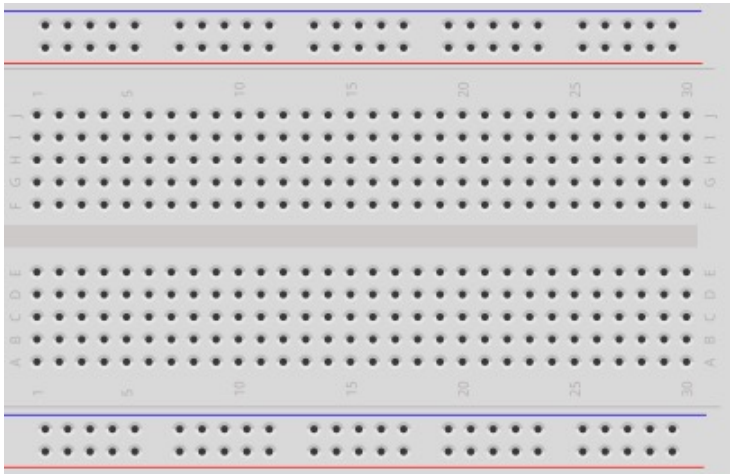
- So far, we have just used the Arduino itself
- Typically, we want to connect external components like LEDS, Temperature Sensors, etc.
- Let's start by using and program an external LED

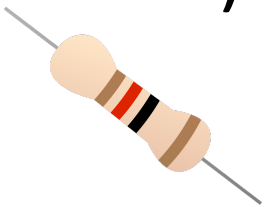
External LED Example

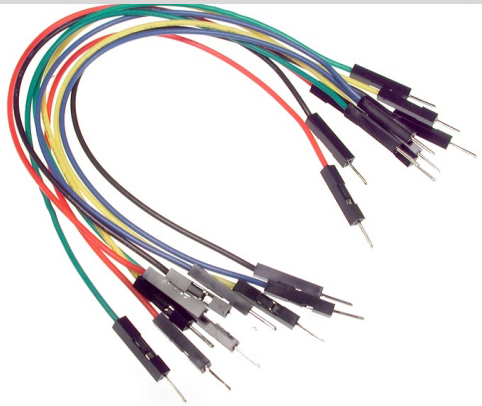
What do we need?

- Breadboard
- LED
- Wires
- Resistor (e.g., $R = 270\Omega$)

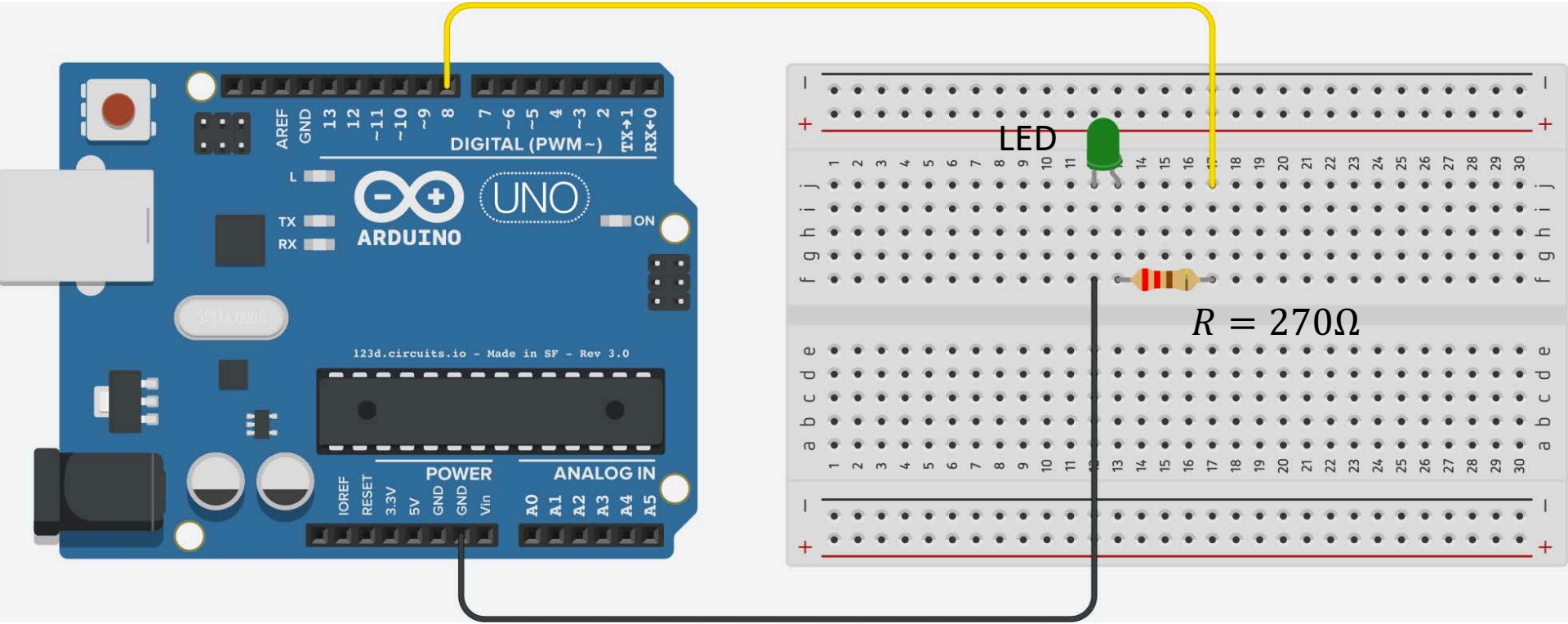




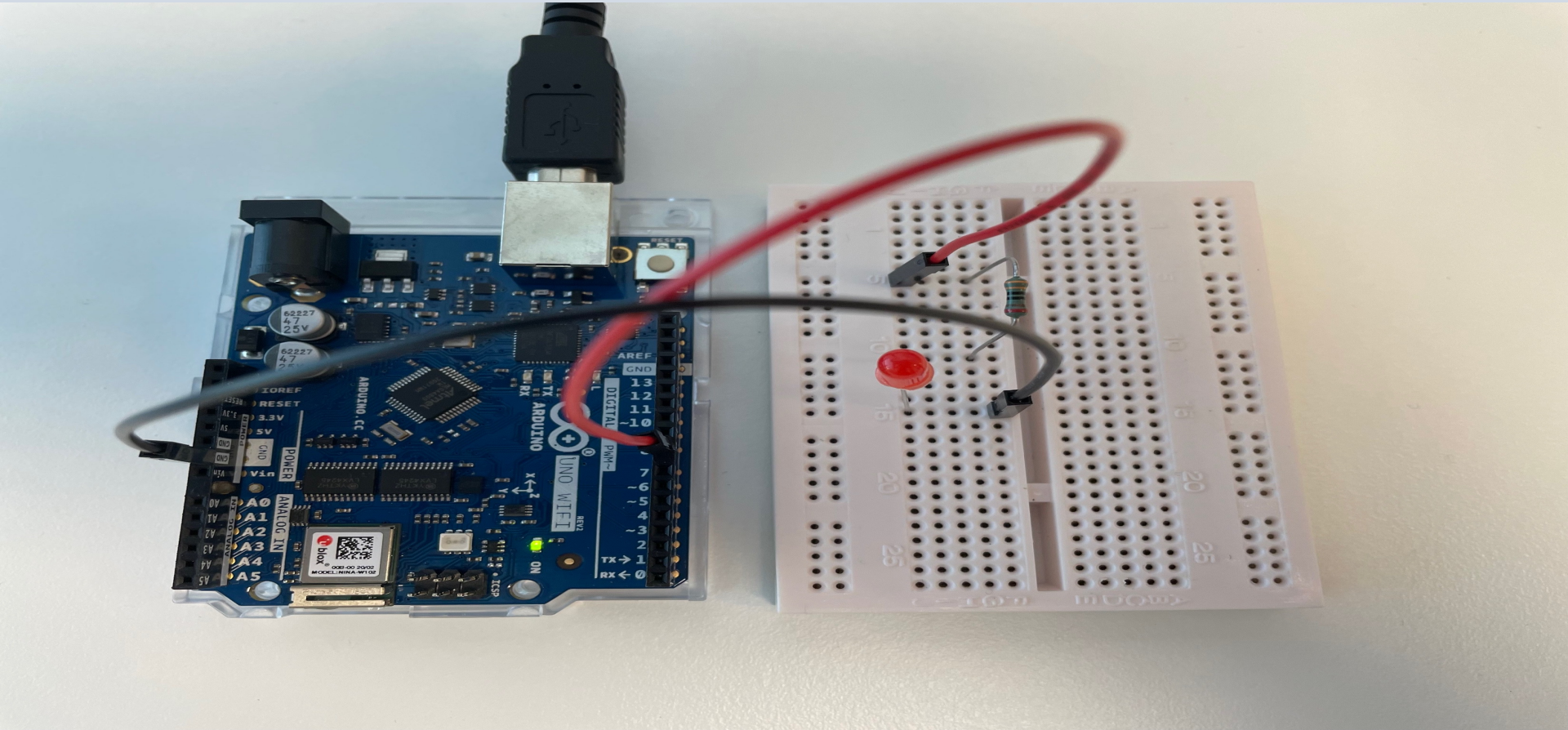




Wiring


LED
$R = 270\Omega$
DIGITAL (PWM~)
ARDUINO
UNO
POWER
ANALOG IN
123d.circuits.io - Made in SF - Rev 3.0

Wiring



Code

```
int ledPin = 8;
int timerWait = 1000;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  digitalWrite(ledPin, HIGH);
  delay(timerWait);
  digitalWrite(ledPin, LOW);
  delay(timerWait);
}
```

The code is the same as for the interna LED, you just need to change the pin number

Functions

Hans-Petter Halvorsen

Table of Contents

Create Functions

- So far, we have used built-in functions like digitalWrite(), delay(), etc.
- Like other Programming Languages it is also possible to create and use your own Functions
- Let's “improve“ the LED example by creating some Functions

Code

```
int ledPin = 8;
int timerWait = 1000;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  ledon();
  delay(timerWait);
  ledoff();
  delay(timerWait);
}

void ledon(){
  digitalWrite(ledPin, HIGH);
}

void ledoff(){
  digitalWrite(ledPin, LOW);
}
```

Self-made Functions

Serial Monitor

Hans-Petter Halvorsen

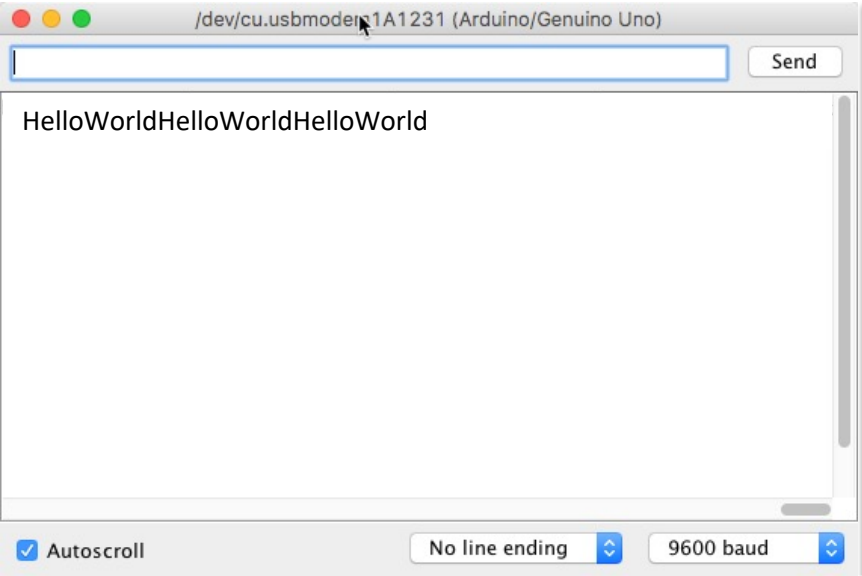
Table of Contents

Serial Monitor

- The Arduino works like an embedded system where you download the code to the device, and then it runs independently of your Computer
- You can remove the USB cable and only connect a Power Supply (or using a 9V Battery)
- This means an Arduino application has no Graphical User Interface and you cannot use a Mouse or a keyboard to communicate with the program

Serial Monitor

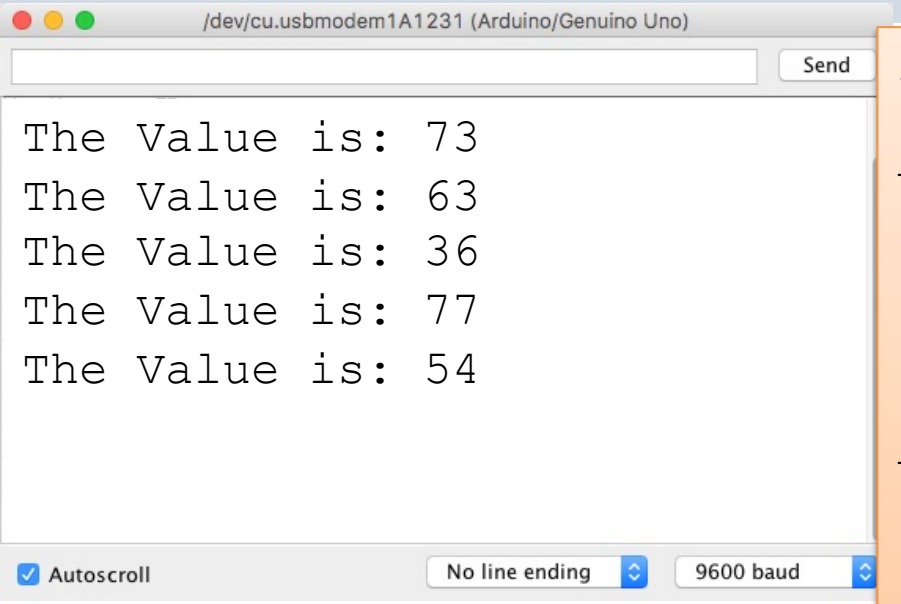
You use the Serial Monitor when **Debugging** Arduino programs or when you want to **show data or values from your program**. You need to have Arduino connected to your PC (using the USB cable) in order to use the Serial Monitor.



`Serial.print()`
`Serial.println()`

```
void setup()
{
        Serial.begin(9600);
}

void loop()
{
        Serial.print("Hello World");
        delay(1000);
}
```

Serial Monitor Example




```
int myValue = 0;

void setup()
{
        Serial.begin(9600);
}

void loop()
{
    myValue = random(100);
    Serial.print("The Value is: ");
    Serial.println(myValue);
    delay(1000);
}
```

Here you see how we can write a value to the Serial Monitor. This can be a value from a sensor, e.g., a temperature sensor.

Example

This Example uses both built-in functions in addition to self-made functions. The results are written to the Serial Monitor

```
int z;int a;int b;
void setup()
{
        Serial.begin(9600);
}
void loop()
{
        a = random(100);
        b = random(100);
        z = calculate(a,b); //Adding 2 Numbers

        //Write Values to Serial Monitor
        Serial.print(a);
        Serial.print(" + ");
        Serial.print(b);
        Serial.print(" = ");
        Serial.println(z);

        delay(1000);
}
float calculate(int x, int y)
{
        return (x + y);
}
```

Sensors and Actuators

Hans-Petter Halvorsen

Table of Contents

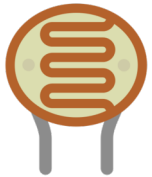
Sensors and Actuators

Temperature sensor

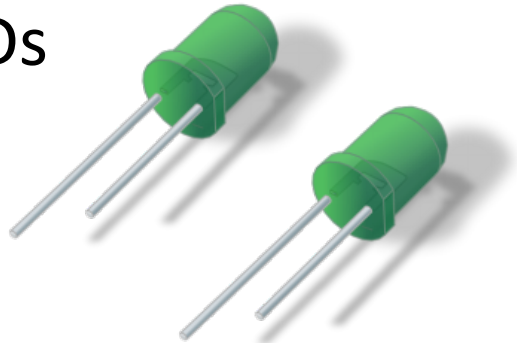
Switch

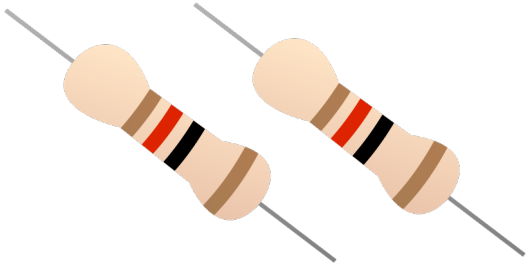
Potentiometer

Light sensor



LEDs





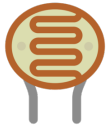
Thermistor

Sensors and Actuators

Temperature Sensor

Push Button

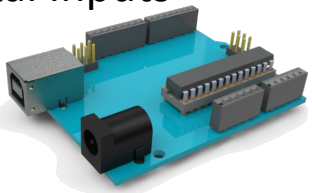
Light Sensor



Potentiometer

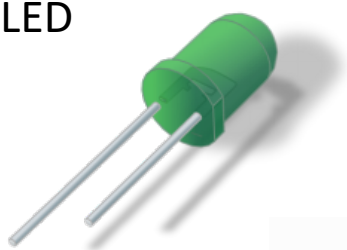
Analog/Digital Inputs

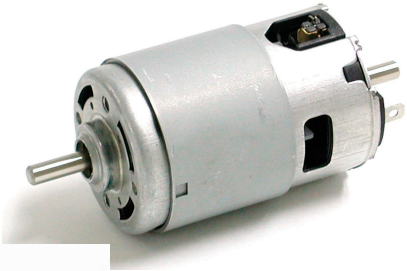
Sensors



Signals from the surroundings

LED

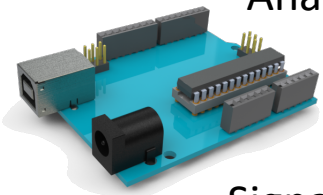




Motor

Buzzer

Analog/Digital Outputs



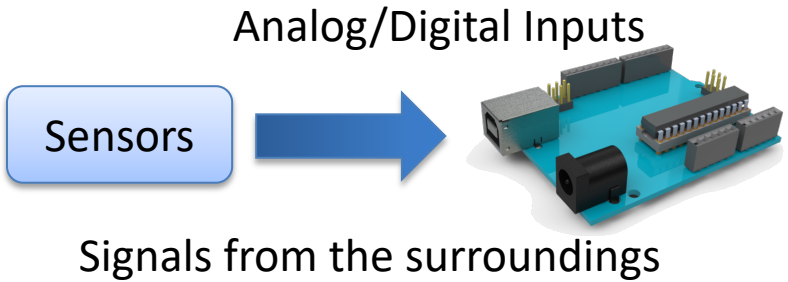
Actuators

Signals to the surroundings

Sensors

A Sensor is a converter that measures a physical size and converts it to a signal that can be read by an instrument, data acquisition device, or an Arduino in our case

Examples: Temperature sensor, Pressure sensor, etc.



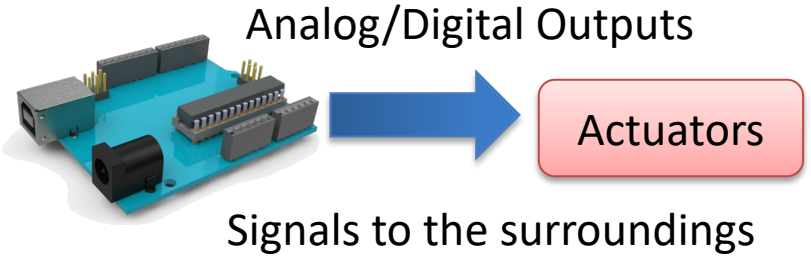
We use **Analog In** pins and **Digital In** pins for reading data from Sensors into the Arduino

Actuators

An Actuator is a kind of motor that moves or controls a mechanism or system.

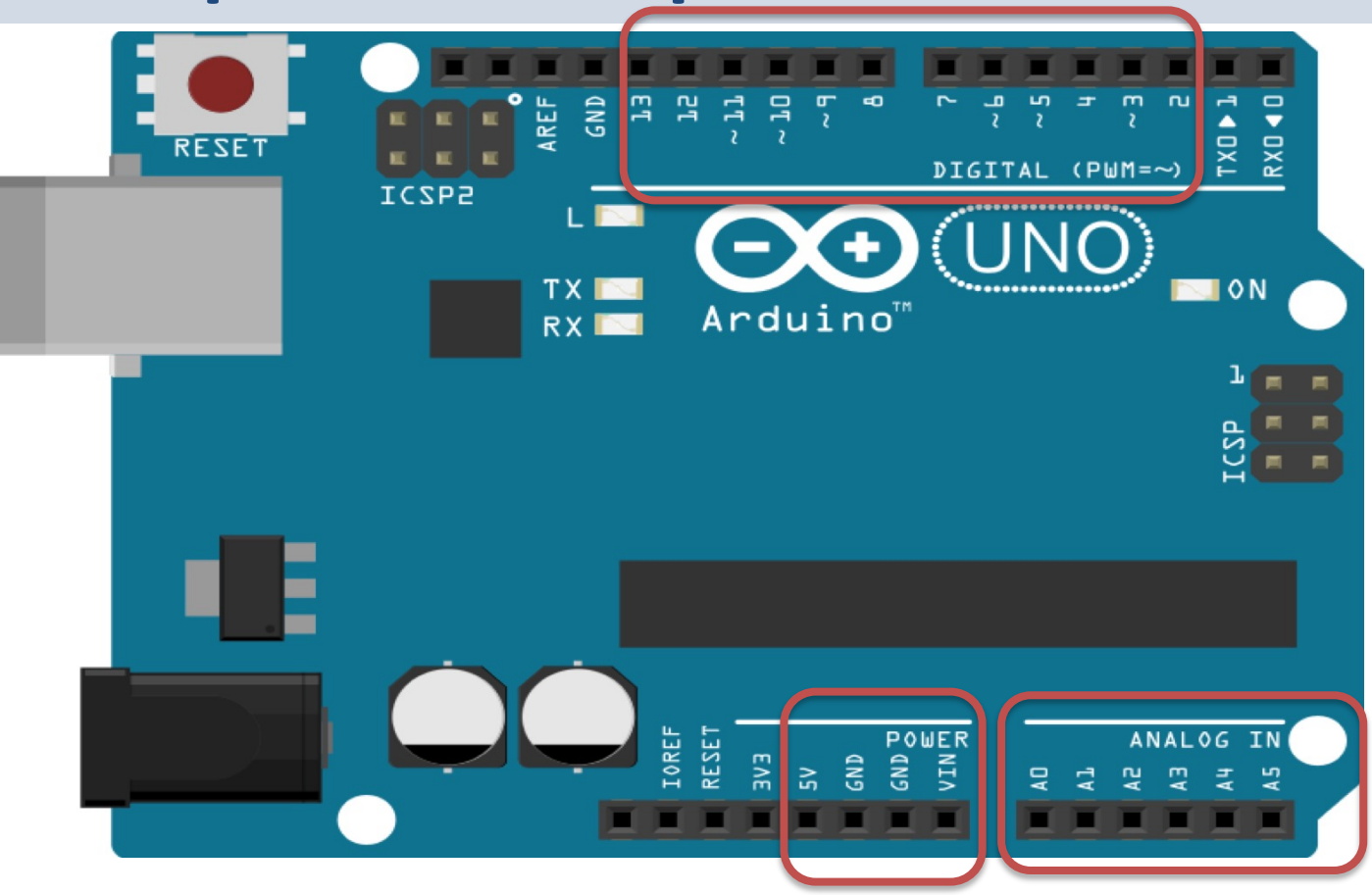
It is powered by an energy source, typical electric current, hydraulic fluid pressure, or air pressure, and converts this energy into motion.

Examples: Engine, Pump, Valve, etc.



We use **Digital Out** pins for controlling the Actuators from the Arduino.
Note! Arduino UNO has no Analog Out pins

Input/Output Pins on Arduino



The Digital pins can be set to be either Digital In or Digital Out. This is done in your Arduino program

Temperature Sensors

Hans-Petter Halvorsen

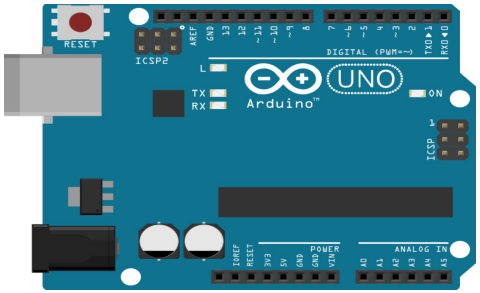
Table of Contents

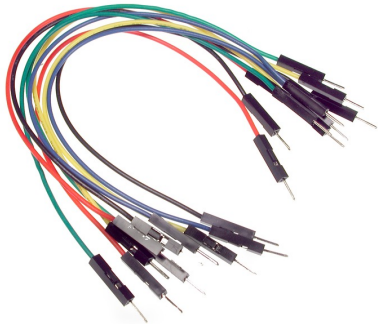
Temperature Sensor Example

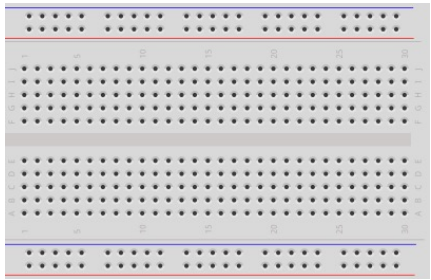
- In this example we will use a small temperature sensor to read the temperature in the room.
- The Temperature Sensor is called “TMP36”
- In this example we will use one of the "Analog In" ports on the Arduino board

Nesessary Equipment

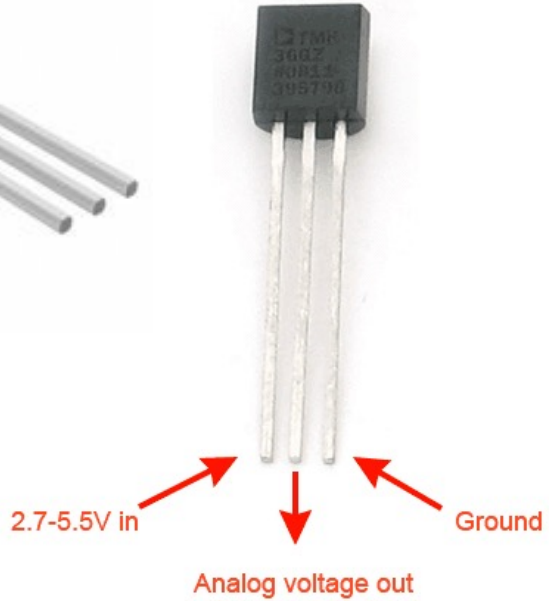
- Arduino
- Breadboard
- TMP36
- Wires (Jumper Wires)

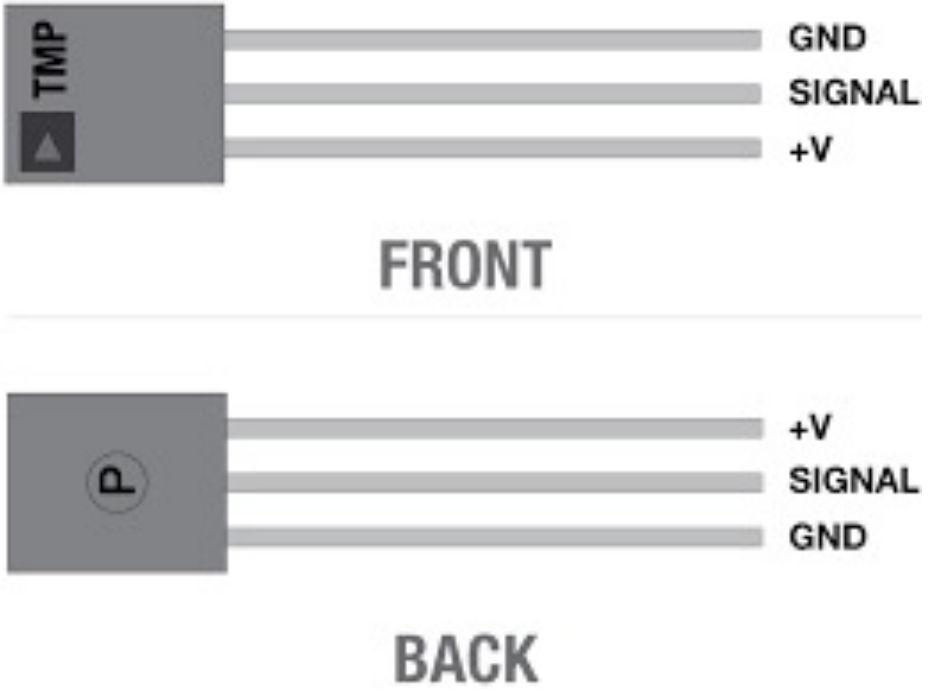






TMP36

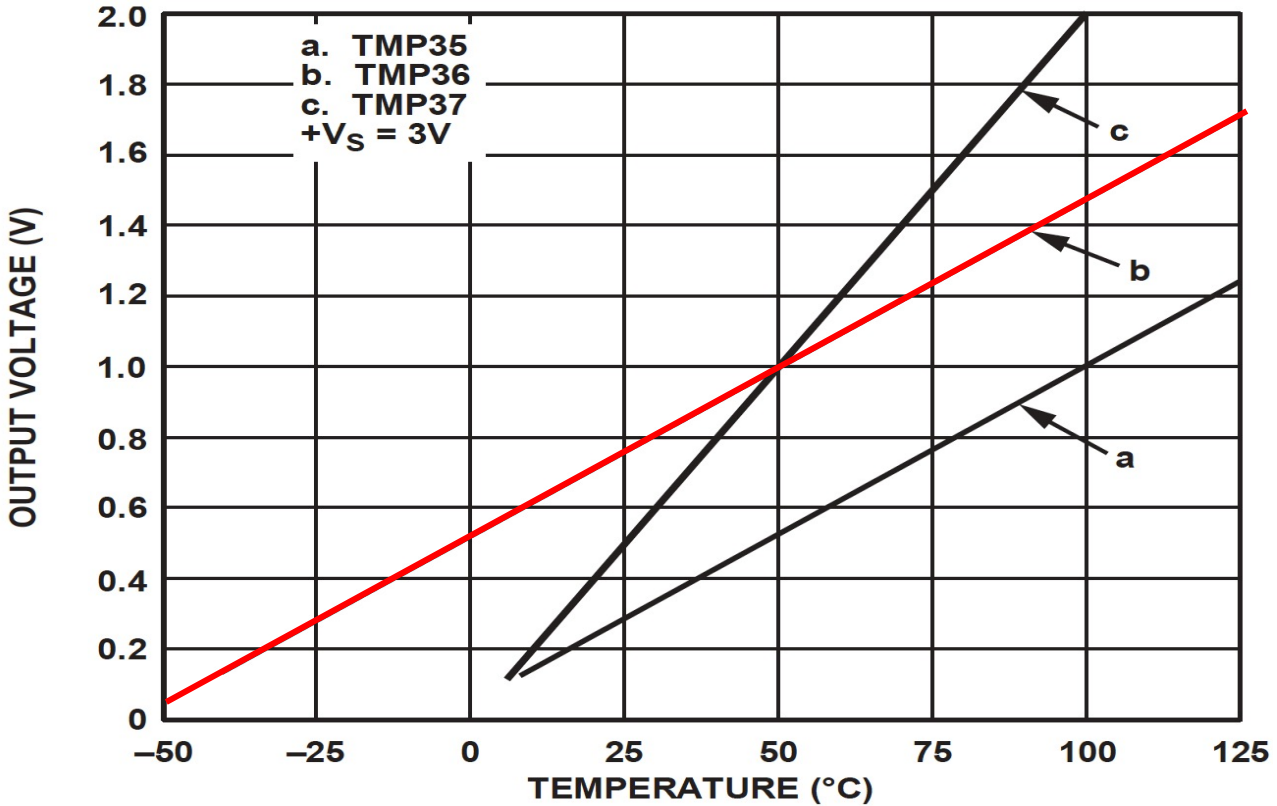




TMP is a small, low-cost temperature sensor and cost about $1 (you can buy it “everywhere”)

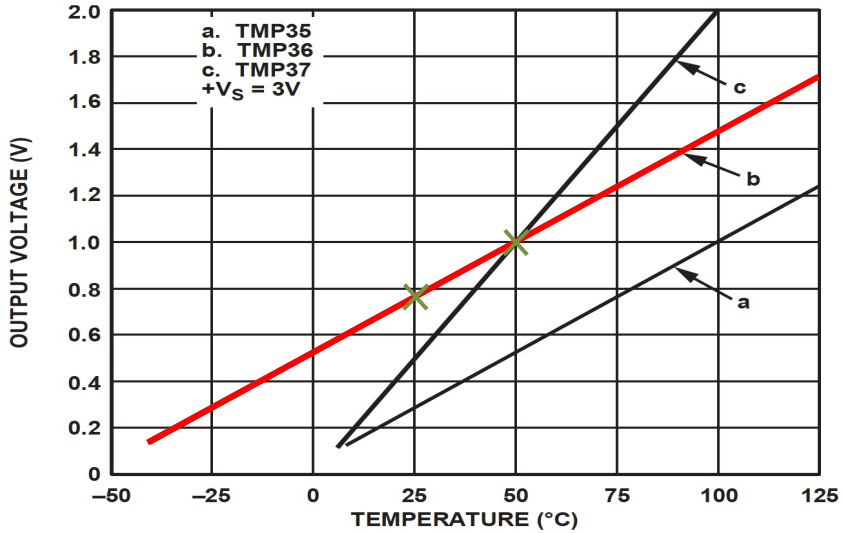
Datasheet

Output Voltage vs. Temperature



https://www.arduino.cc/en/uploads/Main/TemperatureSensor.pdf

Linear Scaling



This gives:

$$y - 25 = \frac{50 - 25}{1 - 0.75}(x - 0.75)$$

Then we get the following formula:

$$y = 100x - 50$$

Convert form Voltage (V) to degrees Celsius
From the Datasheet we have:

$$(x_1, y_1) = (0.75V, 25°C)$$
$$(x_2, y_2) = (1V, 50°C)$$

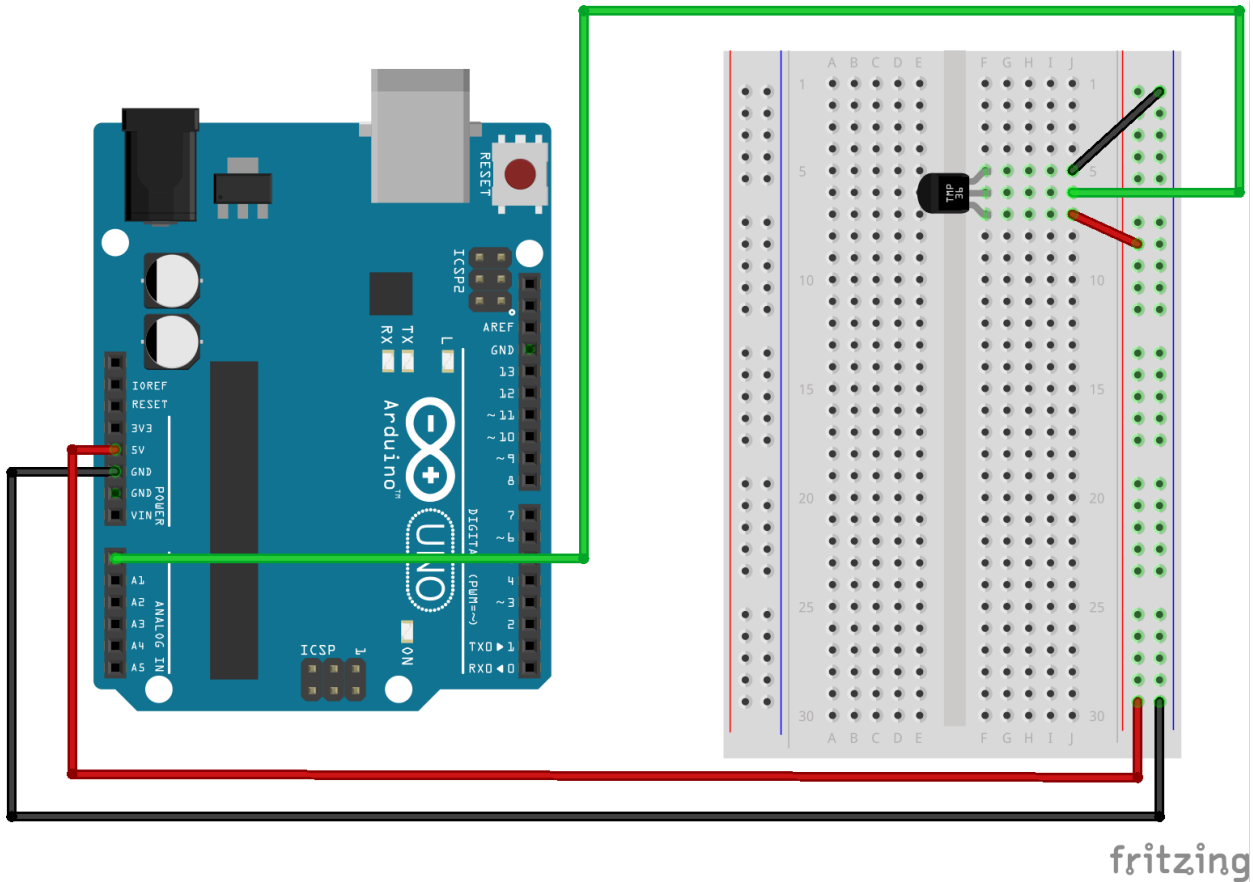
There is a linear relationship between Voltage and degrees Celsius:

$$y = ax + b$$

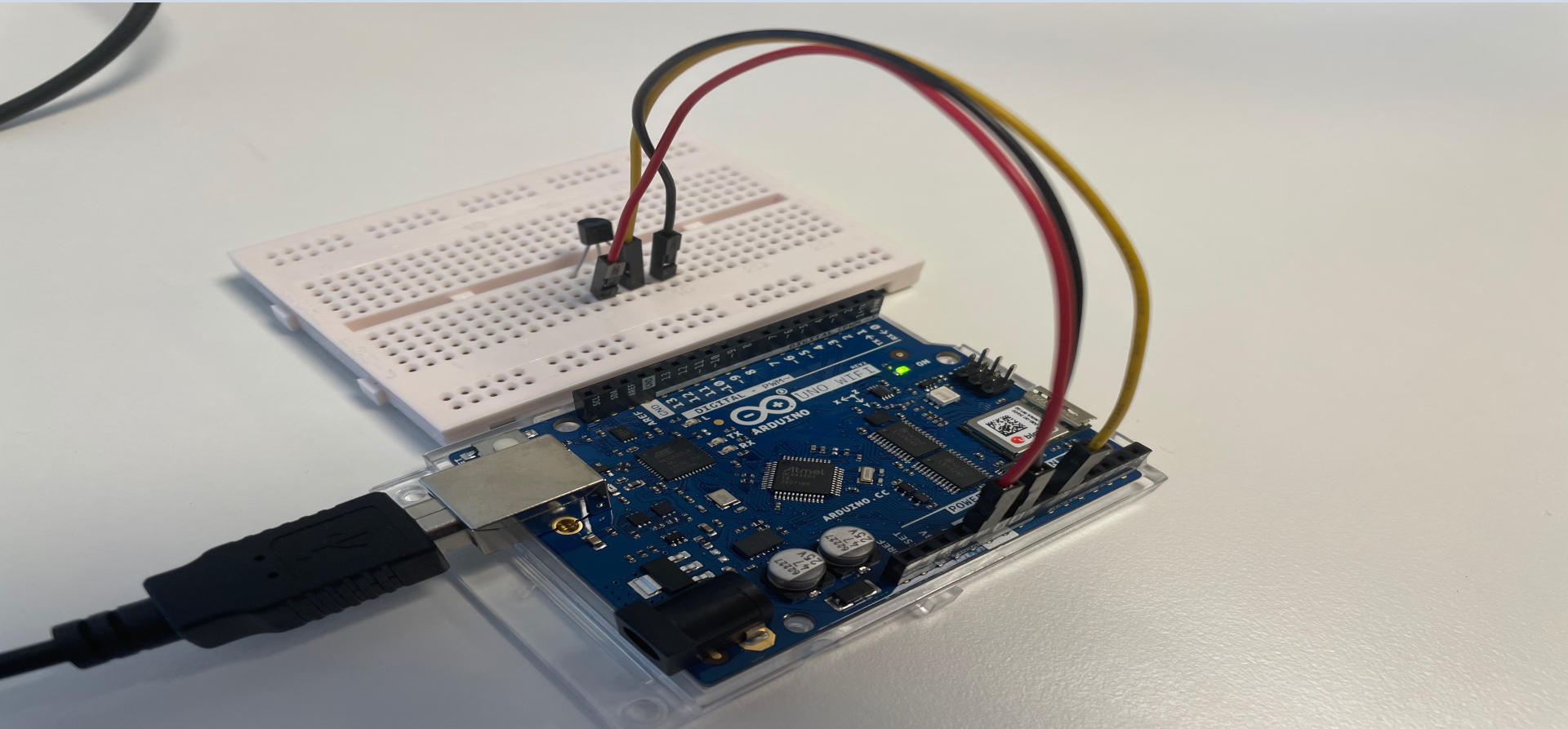
We can find a and b using the following known formula:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

Wiring


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
4
~3
2
TXD▶1
RXD◀0
DIGITAL (PWM=~)
TX
RX
L
Arduino™
UNO
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A1
A2
A3
A4
A5
ANALOG IN
A B C D E
F G H I J
1
5
10
15
20
25
30
fritzing

Wiring



Temperature Conversion

We want to present the value from the sensor in degrees Celsius:

1. The function analogRead() gives a value between 0 and 1023 (Arduino UNO has a built-in 10-bit ADC, 2^10=1024)
2. Then we convert this value to 0-5V.
3. Finally, we convert to degrees Celsius using information from the Datasheet presented on the previous page ($y = 100x - 50$)
4. The we can, e.g., show the Temperature value in the Serial Monitor

Code

```
const int temperaturePin = 0;

float adcValue;
float voltage;
float degreesC;

void setup()
{
  Serial.begin(9600);
}

void loop()
{

  adcValue = analogRead(temperaturePin);

  voltage = (adcValue*5)/1023;

  degreesC = 100*voltage - 50;

  Serial.print("ADC Value: ");
  Serial.print(adcValue);

  Serial.print("  voltage: ");
  Serial.print(voltage);

  Serial.print("  deg C: ");
  Serial.println(degreesC);

  delay(1000);
}
```

https://www.halvorsen.blog

Arduino Programming

Hans-Petter Halvorsen

Table of Contents

Arduino Programming

- We have already created and used Variables
- We have also created and used Functions
- Basically, Arduino Programming is very similar to other Programming Languages, so we can also use For Loops, While Loops, create and use Arrays, If..Else, etc.

https://www.arduino.cc/reference/en/

For Loop

In this program we use a For Loop to find the Sum of 100 Random Numbers.

Then we find the Average.

The Sum and Average are written to the Serial Monitor.

```
int x; int sum = 0; float average = 0;
void setup()
{
        Serial.begin(9600);
}
void loop()
{
    sum = 0;
    for (int i = 0; i<100; i++)
    {
        x = random(100);
        sum = sum + x;
    }

    average = sum / 100;
    Serial.print(" Sum = ");
    Serial.print(sum);
    Serial.print(" , Average = ");
    Serial.println(average);
    delay(1000);
}
```

While Loop

In this program we use a While Loop to find the Sum of 100 Random Numbers.

Then we find the Average.

The Sum and Average are then written to the Serial Monitor.

```
int x; int sum = 0; float average = 0; int i;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  sum = 0;
  i=0;
  while (i<100)
  {
    x = random(100);
    sum = sum + x;
    i++;
  }

  average = sum / 100;
  Serial.print(" Sum = ");
  Serial.print(sum);
  Serial.print(" , Average = ");
  Serial.println(average);
  delay(1000);
}
```

Arrays

```
const int arraysize = 100;
int x;
int sum = 0;
float average = 0;
int myarray[arraysize];

void setup()
{
              Serial.begin(9600);
}

void loop()
{
       sum = 0;
       for (int i = 0; i < arraysize; i++)
       {
              x = random(200);
              myarray[i] = x;
       }
       sum = calculateSum(myarray);
       average = sum / 100;
       Serial.print(" Sum = ");
       Serial.print(sum);
       Serial.print(" , Average = ");
       Serial.println(average);
       delay(1000);
}

int calculateSum (int sumarray[])
{
       for (int i = 0; i < arraysize; i++)
       {
              sum = sum + sumarray[i];
       }
       return sum;
}
```

If .. Else

```
int number;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  number = random(0,100);

  if (number < 50)
  {
     Serial.println("Small Number");
  }
  else
  {
     Serial.println("Large Number");
  }
  delay(1000);
}
```

Summary

- In this Tutorial an overview of Arduino is given
- Arduino is a powerful and flexible device that can be used in many Internet of Things projects
- It is only your imagination that sets the limit
- Good luck with your Arduino projects

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog