

# Werken met Arduino

## 1. Introductie

### 1.1. Wat is Arduino?

Arduino is de naam van een microprocessorbord. Een microprocessor is een kleine computerchip die geprogrammeerd kan worden. Je kunt met een Arduino allerlei apparaten maken. Een robotje, een elektronische dobbelsteen, een looplicht, etc. Er zijn veel verschillende Arduino's. Sommige groot de andere klein. De eigenlijke chip is piepklein, de dingen die verder op het bord staan zoals een usb-aansluiting, in en uitgangen, etc. maken het geheel groter.



Een Arduino programmeer je met een pc, meestal met een laptop. Eerst schrijf je een programma, daarna stuur je het naar de Arduino via een USB Kabel. Als het allemaal goed werkt kun je daarna de kabel weghalen en werkt jouw Arduino project op zichzelf. Heel vaak is het zo dat je een deel van een oude code kan hergebruiken voor andere toepassingen.

### 1.2. Arduino en externe voeding

Als je het programma op je Arduino hebt gezet kan je de Arduino op zichzelf laten werken. Dit wilt zeggen dat je de microprocessor niet meer moet verbinden met je pc of laptop via een usb kabel. Na het aansluiten van een externe voeding zal het laatste geladen programma automatisch starten. Zo'n externe voeding kan een 9Volt batterij zijn of een andere voedingsbron.

Deingangsspanning van een Arduino ligt tussen de 6 en 20 Volt. 4 batterijen van 1,5 Volt in serie gebruiken, is echter geen goed idee. Je bekomt dan net 6 Volt ( $4 \times 1,5 = 6$ ) wat in principe genoeg is, maar als je dan iets wilt schakelen wat een beetje stroom vraagt, zoals een relais, zal de voedingsspanning van de Arduino al snel onder de 5 Volt duiken en gaan sommige componenten vreemd of zelfs niet reageren. Gebruik dus minstens 7 Volt om zeker zo'n situaties te vermijden. Een Arduino heeft een ingebouwd voedingscircuit dat ervoor zorgt dat ongeacht deingangsspanning (6 Volt of er boven) er altijd een stabiele 5 Volt beschikbaar voor de uitgangen.

Hiervan moet je zeker onthouden

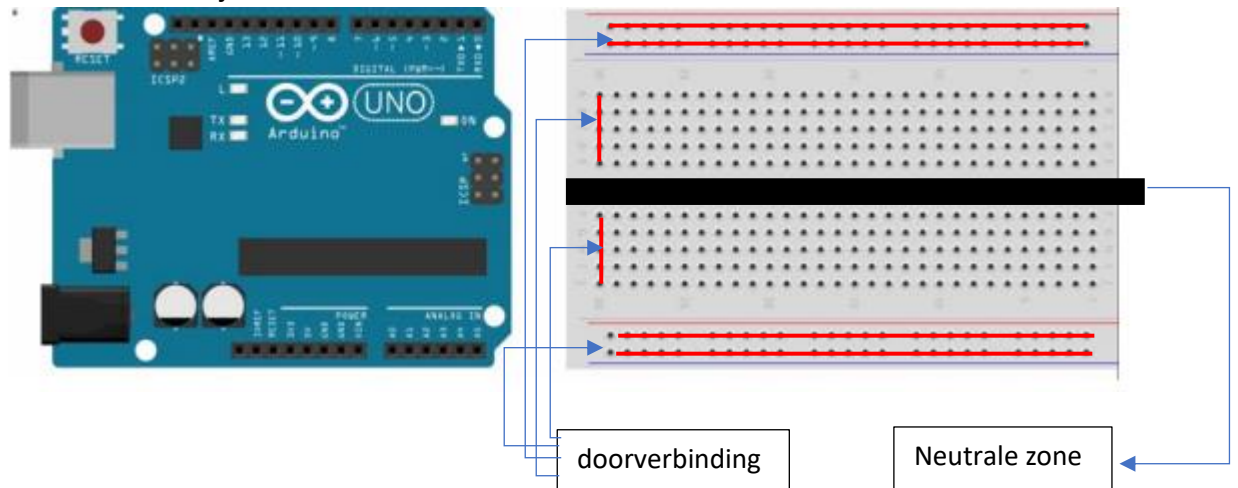
- Voedingsspanning ofingangsspanning van de Arduino bedraagt tussen de 6 en 20 Volt
- Uitgangsspanning +-5 Volt



- Is deingangsspanning van de Arduino lager dan 6Volt, dan gaat de Arduino vreemd functioneren.

### 1.3. Het Breadboard

Om onderdelen op een Arduino aan te sluiten gaan gebruik je best een breadbord. Een breadboard is paneel met allemaal kleine gaatjes in. In die gaatjes kan je draden of componenten steken om verbindingen te maken. De bovenste en onderste twee rijen vormen horizontale verbindingen de rest is verticaal doorverbonden. Met een neutrale zone in het midden waar geen verbindingen doorlopen. De bovenste en onderste helft zijn dus niet met elkaar verbonden.



Verbindingen maken tussen de arduino en het breadbord doen we door gebruik te maken van 'jumpers'. Dat zijn de kabeltjes met verschillende kleuren. Je hebt verschillende soorten kabels, zo zijn er 'male-male', 'male-female' en 'female-female'. Een 'male' of mannelijke connector is er één die je ergens in moet steken, een vrouwelijke is de connector waar je iets in moet steken.

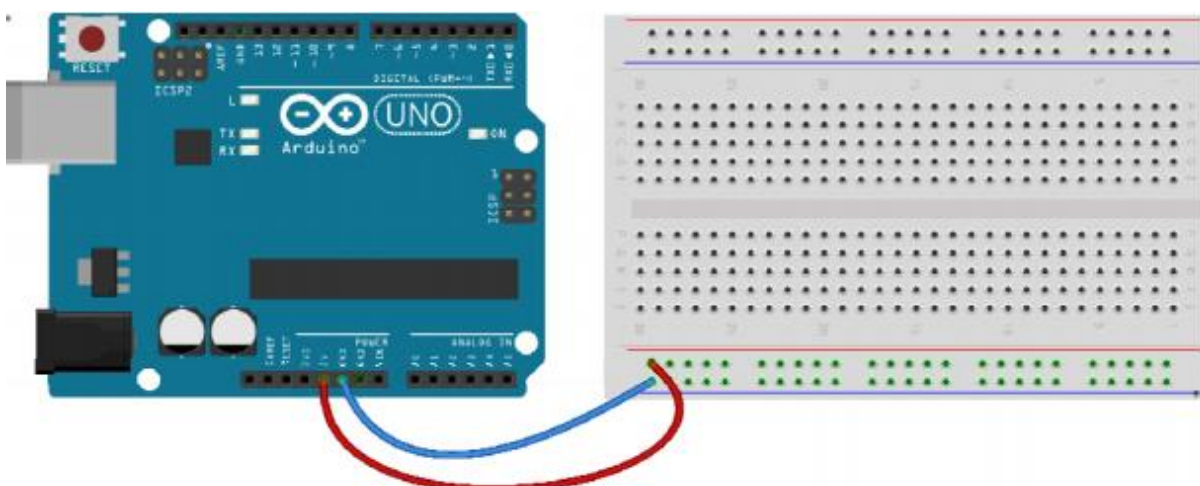


*Figuur 1 male-male jumpers*

Het breadboard krijgt spanning via de Arduino, die op zijn beurt voedingsspanning krijgt van de PC/laptop of externe bron. We verbinden de Gnd (ground of de -) en de 5V aansluitingen van de Arduino met een blauwe (Gnd) en rode (5V) draad aan het breadboard. Gebruik altijd een rode draad voor de + en een blauwe draad voor de -.

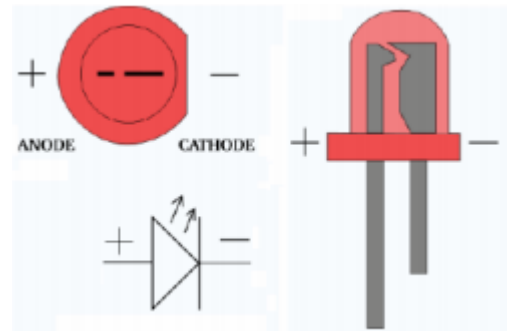


*Figuur 2 male-female*



## 1.4. LED's

Een Led is een light emitting diode, vertaald: een licht uitstralende diode. Die werkt heel anders dan een normale gloeilamp. Een Led heeft een + (anode) en – (cathode) kant. Het lange beentje is de +, het korte beentje is de -. Zijn de beentjes dezelfde lengte dan kan je nog altijd kijken naar de led zelf. De – is de platte kant van de led.



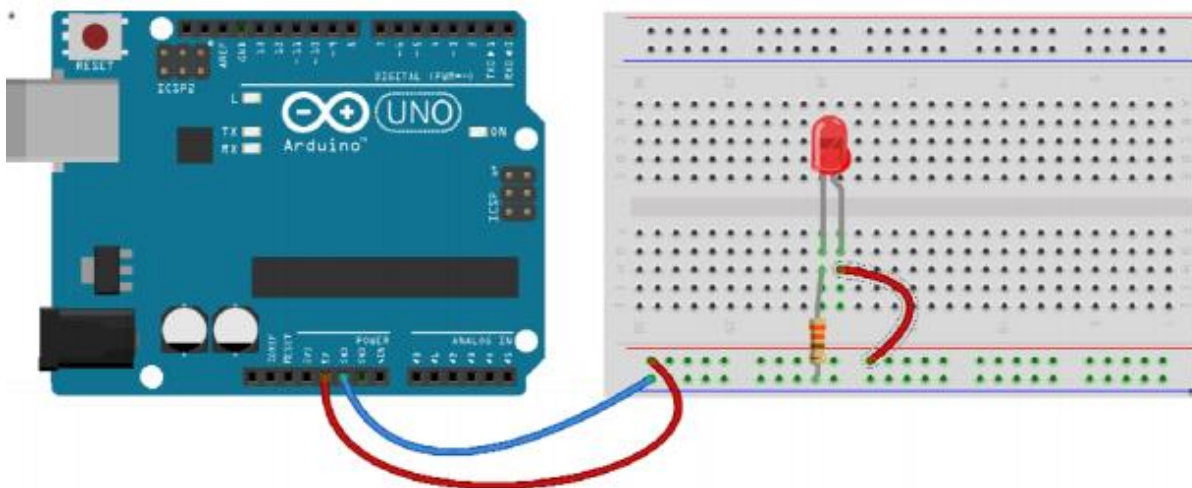
Led's kunnen doorbranden. Daarom wordt er altijd een weerstand in serie met de led geplaatst die de stroom gaat beperken. Het maakt niet uit of de weerstand aan de + of – kant gezet wordt. Om te berekenen hoe groot de voorschakelweerstand moet zijn, ga je die moeten berekenen aan de hand van volgende formule:  $R = (U_b - U_{led}) / I$ . Dit wilt zeggen dat je de bronspanning – de spanning die de led vraagt deelt door de stroom. Het getal dat we dan uitkomen is de waarde die de weerstand minimaal moet hebben. Bestaat de uitgekomen weerstandswaarde niet of is het kommagetal neem dan altijd een weerstandswaarde hoger. **Een rode led heeft altijd een voedingspanning van 1,9Volt.**

### Voorbeeld:

$$R = (U_b - U_{led}) / I$$

$$R = (5 - 1,9) / 0,02 = 3,1 / 0,02 = 155 \text{ Ohm}$$

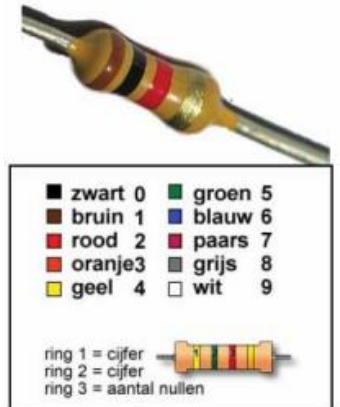
Je neemt gemakshalve een weerstandswaarde van 180 ohm (een algemeen verkrijgbare weerstandswaarde) als voorschakelweerstand.



Als je de schakeling maakt zoals afgebeeld gaat de led branden zodra je de Arduino op de pc aansluit.

### 1.5. Weerstanden

De waarde van een weerstand wordt altijd afgebeeld in kleuren op de weerstand zelf en uitgedrukt in ohm. De gekleurde ringen op een weerstand zijn hiervan een indicator. De weerstand die hiernaast staat afgebeeld is er één van 1000ohm (1K ohm) want bruin is **1**, zwart is **0** en rood geeft het aantal nullen weer en dat is **2**. De vierde ring geeft de tolerantie weer. Goud staat voor een 5% tolerantie. Tolerantie wilt zeggen dat de weerstandswaarde, mochten we die gaan meten, een beetje afwijkt van de waarde berekend aan de hand van de kleuren.



### 1.6. Aansluitingen maken met de PC en laptop

Nu we de basis door hebben kunnen we beginnen met het programmeren, maar vooraleer we kunnen programmeren moeten we het Arduino programma downloaden en installeren van de Arduino website. Dat doe je best via de Arduino website zelf. Aangezien de software toch gratis is, zijn er geen nadelen aan verbonden.

Website: <https://www.arduino.cc/en/main/software> als je op Windows werkt download dan best de Windows versie.

**ARDUINO 1.8.8**  
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.  
This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions.

**Windows** Installer, for Windows XP and up  
**Windows** ZIP file for non admin install

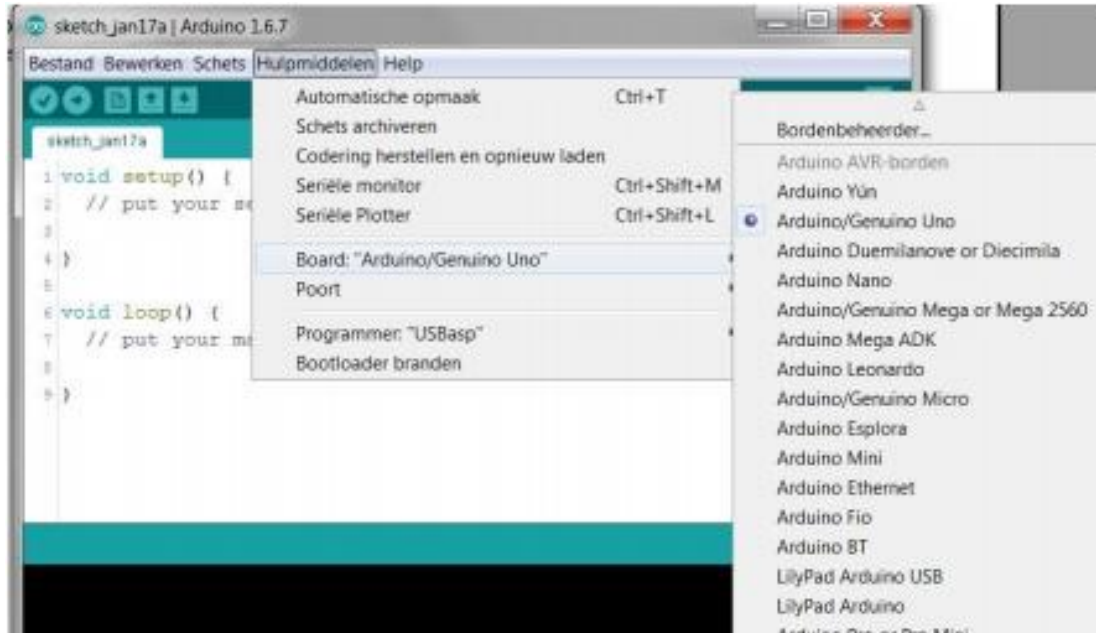
**Windows app** Requires Win 8.1 or 10  
[Get](#)

**Mac OS X** 10.8 Mountain Lion or newer

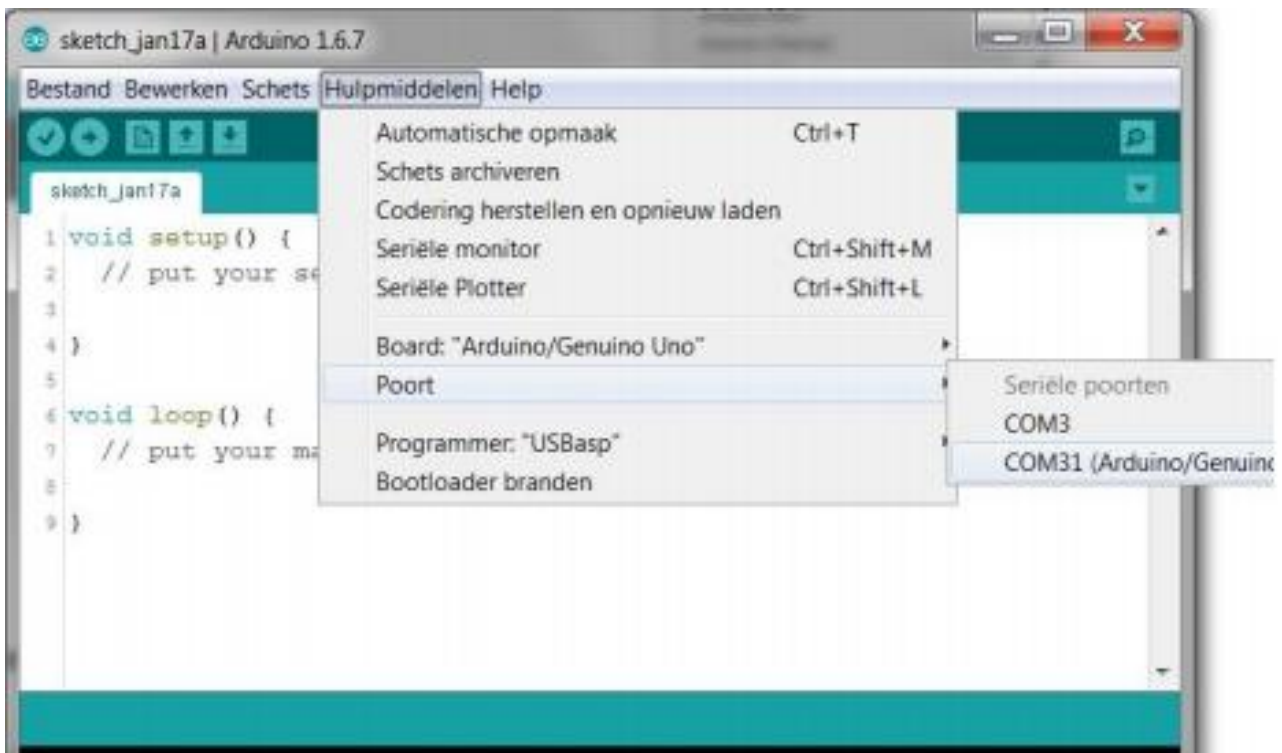
**Linux** 32 bits  
**Linux** 64 bits  
**Linux** ARM

[Release Notes](#)  
[Source Code](#)  
[Checksums \(sha512\)](#)

Als het Arduino programma geïnstalleerd is op je pc kan je de Arduino aansluiten met een usb aan diezelfde pc. Open de Arduino software en klik op 'hulpmiddelen'. Kies dan als Board voor een Arduino/Genuino Uno omdat dat het bord is dat we ook in dit project gebruiken.



Blijf in de tap 'Hulpmiddelen' en ga naar 'Poort' controleer of de pc de Arduino kan zien. Als dat niet zo is, controleer dan of de Arduino goed is aangesloten op je pc. Als het dan nog niet lukt, ga dan kijken bij 'apparaat beheer' op je pc of laptop op welke com poort je Arduino is aangesloten.



## 2. Programmeren

We kunnen nu al uit de doeken doen hoe wij ons Arduino programma hebben geschreven, maar dat lijkt ons niet zo verstandig. Als leerlingen dat programma zelf moeten programmeren hebben ze enige voorkennis nodig. Diezelfde voorkennis heb ik meegekregen uit mijn vooropleiding.

Er zijn enkele belangrijke kenmerken van de programmeertaal die in elk programma dat je gaat maken hetzelfde is.

- Variabelen kan je gebruiken om getallen op te slaan in het geheugen om later weer te kunnen gebruiken. Er zijn verschillende types variabelen die meer of minder geheugenplek innemen, afhankelijk van het aantal getallen dat ze moeten opslaan.  
De soort die wij gebruiken is een integer. Int (integer) is het meest gebruikte data type. Het heeft geen decimalen en kan een getal opslaan tussen -32.768 en 32.767 en dat is voldoende voor onze toepassing.
- **Alles tussen /\* en \*/ is commentaar.** Commentaar wordt niet door de Arduino uitgevoerd. Commentaar is handig om jou programma duidelijk te maken aan andere programmeurs.
- **Elke regel die begint met //** is ook Commentaar en wordt niet uitgevoerd door de Arduino. `//declareren van de variabelen`

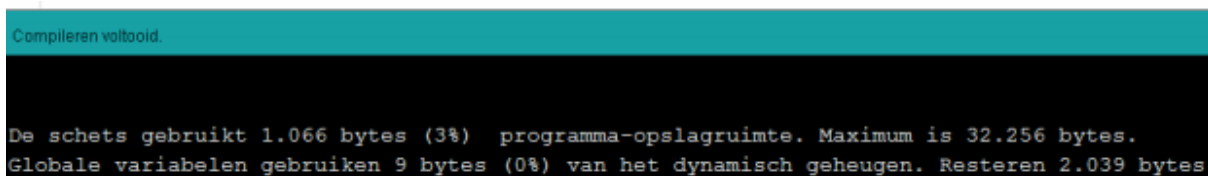
- **Achter elke regel code** (dus niet de commentaren) komt een ; Doe je dit niet dan geeft je programma een fout en wordt het zoeken naar wat je fout gedaan hebt.
- Elke Arduino programma heeft twee blokken een **setup** en een **loop blok**. Zo'n blok begint met een { en eindigt met een }. In het setup blok staan dingen die je éénmaal wilt vastleggen. Bijvoorbeeld dat je een led hebt aangesloten op Pin 13 en je die pin 13 wilt gebruiken als uitgang. De opdracht daarvoor is: `pinMode(13, OUTPUT);`

```
int val;
int tempPin = 0;
int sensorPin = A5;
int sensorWaarde = 0;
int sensorMin = 350;
int sensorMax = 1023;
int Vochtigheid;
```

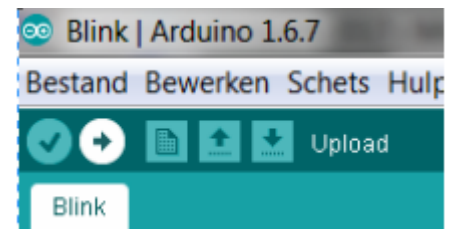
In het **loop** blok staat het programma dat door de Arduino voortdurend doorlopen wordt. Hier komt het eigenlijke programma. Een voorbeeld van code die hier kan komen is :

<code>digitalWrite(13, HIGH);</code>	Hiermee wordt pin 13 'Hoog' gemaakt. Dat is hetzelfde als 'met plus verbonden'. De Led gaat aan. Let op! ELKE regel wordt in Arduino met ; afgesloten!
<code>delay(1000);</code>	De opdracht om 1000 milliseconden helemaal niets te doen
<code>digitalWrite(13, LOW);</code>	Hiermee wordt pin 13 'Laag' gemaakt. Dat is hetzelfde als 'met min verbonden'. De Led gaat uit.
<code>delay(1000);</code>	De opdracht om 1000 milliseconden helemaal niets te doen

- bovenaan in het programeer venster staan twee knoppen die je bij elk programma gaat nodig hebben namelijk de **compileren** en **upload** knop. Compileren controleert of het programma goed geschreven is en er geen fouten gemaakt zijn. Mochten er fouten zijn wordt er een foutmelding weergegeven onderaan het scherm.



De upload knop wordt enkel gebruikt om je programma naar je Arduino te sturen.



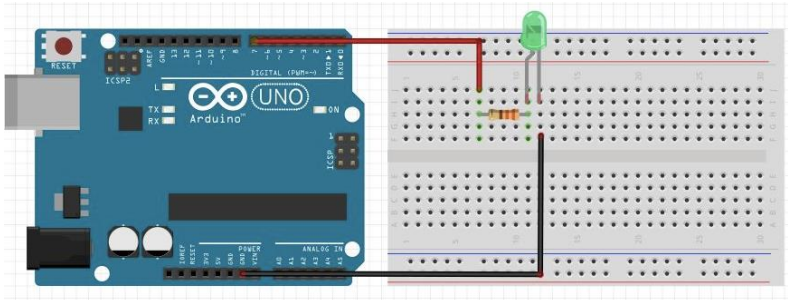
Nu we dit hebben doorlopen kunnen we naar het programmeren zelf gaan.

## 2.1. Instap oefeningen

### Oefening 1

Laat één led flikkeren aan een snelheid van 1 keer per seconde.

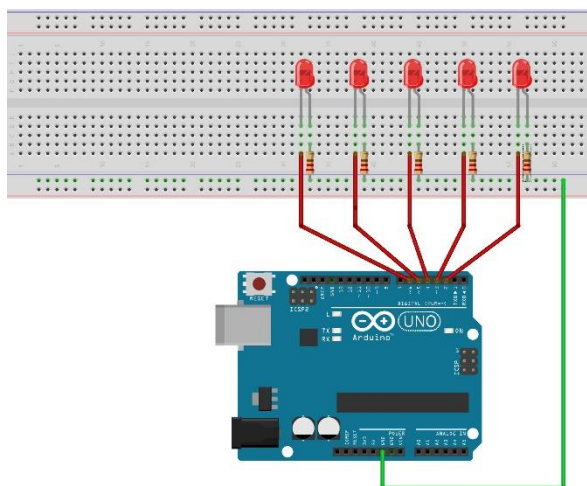
Aansluiting:



### Oefening 2.1

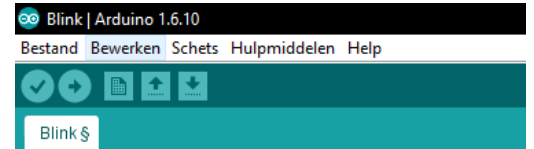
Maak een looplicht van 5 leds elke led brand 1seconde.

Aansluiting:



### Oefening 2.2

Maak een looplicht van 5 leds elke led brand 0,5 seconde.



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

```
void setup() {  
    pinMode(1,OUTPUT);  
    pinMode(2,OUTPUT);  
    pinMode(3,OUTPUT);  
    pinMode(4,OUTPUT);  
    pinMode(5,OUTPUT);  
}
```

```
void loop() {  
    digitalWrite(1,HIGH);  
    delay(1000);  
    digitalWrite(1,LOW);  
    delay(1000);  
    digitalWrite(2,HIGH);  
    delay(1000);  
    digitalWrite(2,LOW);  
    delay(1000);  
    digitalWrite(3,HIGH);  
    delay(1000);  
    digitalWrite(3,LOW);  
    delay(1000);  
    digitalWrite(4,HIGH);  
    delay(1000);  
    digitalWrite(4,LOW);  
    delay(1000);  
    digitalWrite(5,HIGH);  
    delay(1000);  
    digitalWrite(5,LOW);  
    delay(1000);  
}
```

Figuur 3 oefening 2.1

De mogelijkheden zijn eindeloos. Je kan de toepassingen zo gek niet bedenken of je kan het maken met een Arduino. De oefeningen hierboven zijn enkele instapoefeningen die leerlingen kunnen maken om zich vertrouwd te maken het programma



## 2.2. Ons programma

Hieronder staat de code die gebruikt wordt in het project. Op volgende pagina staat er meer uitleg bij de verschillende regels code.

We beginnen bij het begin, het declareren van variabelen. Kortweg betekend dit dat je verschillende woorden gaat gebruiken om waardes in op te slaan.

```
//declareren van de variabelen
int val; // de variabele "val" als een integer.
int tempPin = 0; // tempPin als een integer met waarde 0.
int sensorPin = A5; // Selecteer de analoge ingang.
int sensorWaarde = 0; // Variabele om de sensor waarde op te slaan.
int sensorMin = 350; // Waarde bij 100% vochtigheid
int sensorMax = 1023; // Waarde bij 0% vochtigheid
int Vochtigheid; // de variabele "Vochtigheid" als een integer.
```

daarna gaan we naar de setup

```
void setup()
{
  pinMode(13, OUTPUT); //de led aangesloten op pin 13 als uitgang zetten.
  pinMode(7, OUTPUT); // pin 7 als uitgang programmeren.
  Serial.begin(9600); // de seriële monitor laten beginnen.
}
```

en als laatste de loop

```
void loop()
{
  val = analogRead(tempPin); // de variabele "val" vullen met het analoge signaal
  float mv = ( val/1024.0)*5000; // Het spanningssignaal dat aan de analoge ingang binnen komt omzetten naar een temperatuur.
  float cel = mv/10; // In "cel" gaan we dat getal opslaan.

  Serial.print("TEMPERATUUR = "); // In de seriële monitor wordt aan het begin van elke regel "Temperatuur=" gezet.
  Serial.print(cel); // Achter "TEMPERATUUR=" wordt het getal dat in de variabele "cel" zit gezet.
  Serial.print("°C"); // Na het getal komt "°C".
  Serial.println(); // een enter.
  delay(1000); // Met een vertraging van 1sec wordt er telkens een nieuwe regel geprint.

  sensorWaarde = analogRead(sensorPin); // Lees de waarde van de sensor.
  sensorWaarde = constrain(sensorWaarde, sensorMin, sensorMax); // Stel de grenzen vast van de waarde.
  Vochtigheid = map(sensorWaarde, sensorMin, sensorMax, 100, 0); // Map de sensorwaarde aan een percentagegetal.
  Serial.print("Vochtigheid = "); // Print de waarde van de sensor op de seriële monitor.
  Serial.print(Vochtigheid); Serial.println("%");
  delay(1000);

  if (cel >= 23) // De if functie wordt hier geopend, Als het getal dat in cel zit groter of gelijk is aan 23 er moet er iets gebeuren.
  {
    digitalWrite(LED_BUILTIN, HIGH); // Als het getal/temperatuur groter of gelijk is aan 23 gaat de led branden.
    digitalWrite(7, HIGH);
    digitalWrite(13, HIGH);
  }
  else // Als het getal/temperatuur niet groter of gelijk is aan 23 moet er iets anders gebeuren.
  {
    digitalWrite(LED_BUILTIN, LOW); // Als het getal/temperatuur niet groter of gelijk is aan 23 gaat de led uit.
    digitalWrite(7, LOW);
    digitalWrite(13, LOW);
  }
}
```

### 2.2.1. Waarom gebruiken we deze code?

**De int(integers):** zoals hierboven al vermeld is een integer groot genoeg voor de toepassingen die wij gebruiken.

**pinMode(13,OUTPUT)/(7,OUTPUT):** we zeggen hier dat de alles wat aangesloten is op pin 13 (de led) en pin 7 (de relais) als uitgang wordt aangestuurd.

**Serial.begin(9600):** Met dit commando laten we da Arduino de seriële monitor opstarten. Dat is een scherm waarop verschillende waarden op het computerscherm weergegeven worden. In ons geval de temperatuur en de vochtigheid.

**Val = analogRead(tempPin):** De variable "val" vullen met wat er gelezen wordt in de tempPin, dus ingang 0.

**Float mv = (val/1024.0)\*5000 :** Met dit deel code wordt het getal dat in de variabele "Val" zit gedeeld door 1024.0 en vermenigvuldigd met 5000. Dit nieuw getal wordt in "mv" gezet.

**Float cel =mv/10:** het getal dat in "mv" zit wordt gedeeld door 10 en in "cel" gezet.

**Serial.print("tempratuur = "):** Nu wordt voor elke regel die we in de seriële monitor zien "TEMPRATUUR = " gezet.

**Serial.pint(cel):** Na de "TEMPRATUUR =" wordt de waarde die in cel zit gezet.

**Serialprint("\*C"):** Achter de waarde komt \*C van graden Celsius. Waarom een maalteken en geen "" omdat zo'n symbool niet herkent wordt in de seriële monitor.

**Serialprintln():** er wordt een enter regel geplaatst.

**Delay(1000):** De waarde wordt in de seriële monitor weergegeven één keer per seconde.

Het volgende deel code dient om de vochtigheidssensor te bedienen. Het principe is hetzelfde.

We lezen eerst de analoge waarde van pin A5 in.

**SensorWaarde = constrain(sensorwaarde, sensorMin, sensorMax):** Dit gaat de maximale en minimale waarde van de sensorwaarde vastleggen.

**SensorWaarde = map(sensorwaarde, sensorMin, sensorMax, 100, 0):** We zorgen er nu voor dat de overeen gaat komen met een percentage van 100% tot 0%. Verder gaan we weer de seriële monitor bedienen. Dit doen we op dezelfde manier als bij de temperatuur.

Het volgende deel code is een functie meer bepaald een **“if” functie**. Met een **“Else” uitbreiding**. Kortweg wilt dit deel code zeggen dat als de waarde in Cel groter of gelijk is aan 23 de relais ( pin 7) en de led(pin13) gaan schakelen, maar als er niet aan die voorwaarde voldoet gaat de led en relais niet branden of schakelen. Of ze schakelen uit.

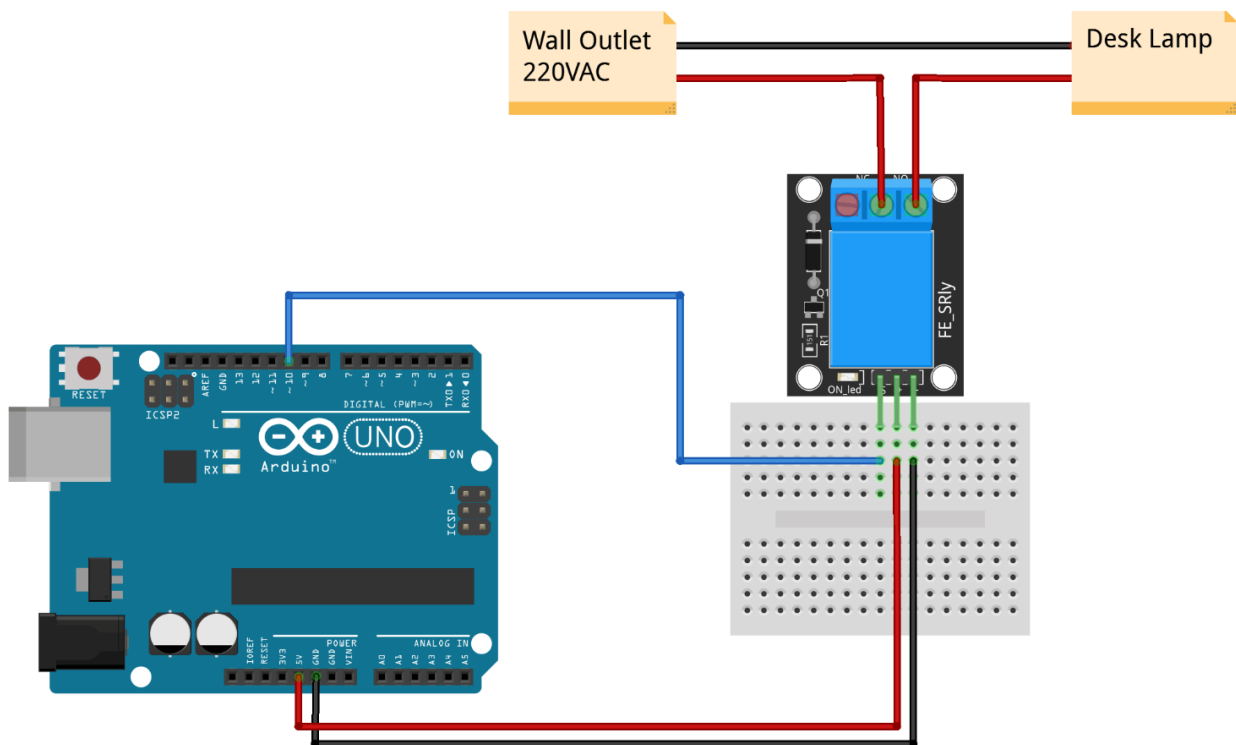
### 3. De aansluitingen

De aansluitingen op de in en uitgangen komen niet overeen met ons programma, maar de tekeningen kunnen perfect dienen als illustratie. Voor een correcte aansluitingspinnen bekijk het programma. **Relais zit op pin 7.**

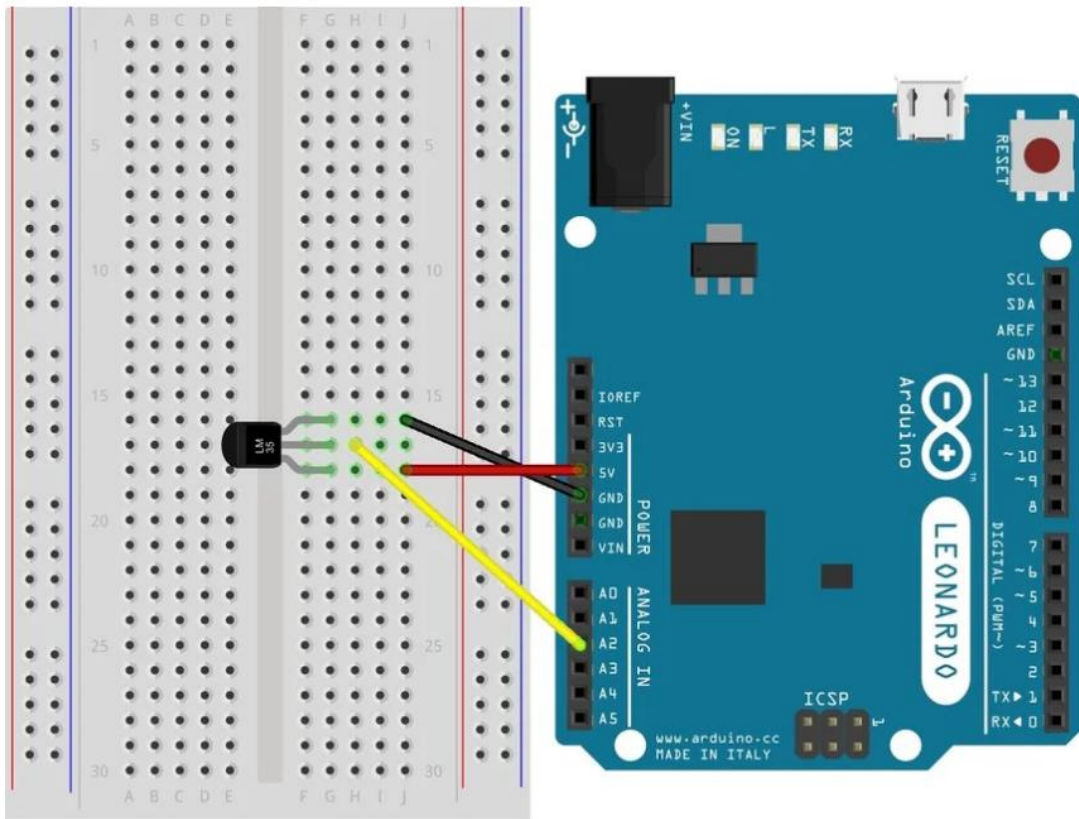
**De led zit op pin 13. De temperatuur sensor op pin A0 en de vochtigheidssensor op pin A5.** Aangezien we maar één van elk hebben, maar wel twee 'kasten' (een leerlingen en een leerkrachten kast) hebben we die twee sensoren gespitst. De temperatuursensor zit op de leerkrachten kast en de vochtigheidssensor op die van de leerlingen. Het principe is hetzelfde voor beide kasten alleen is de kast van de leerlingen kleiner.

Het programma werkt perfect met maar één sensor. Zet het gedeelte van de sensor die niet aangesloten is in commentaar om vreemde waarde in de seriële monitor te vermijden.

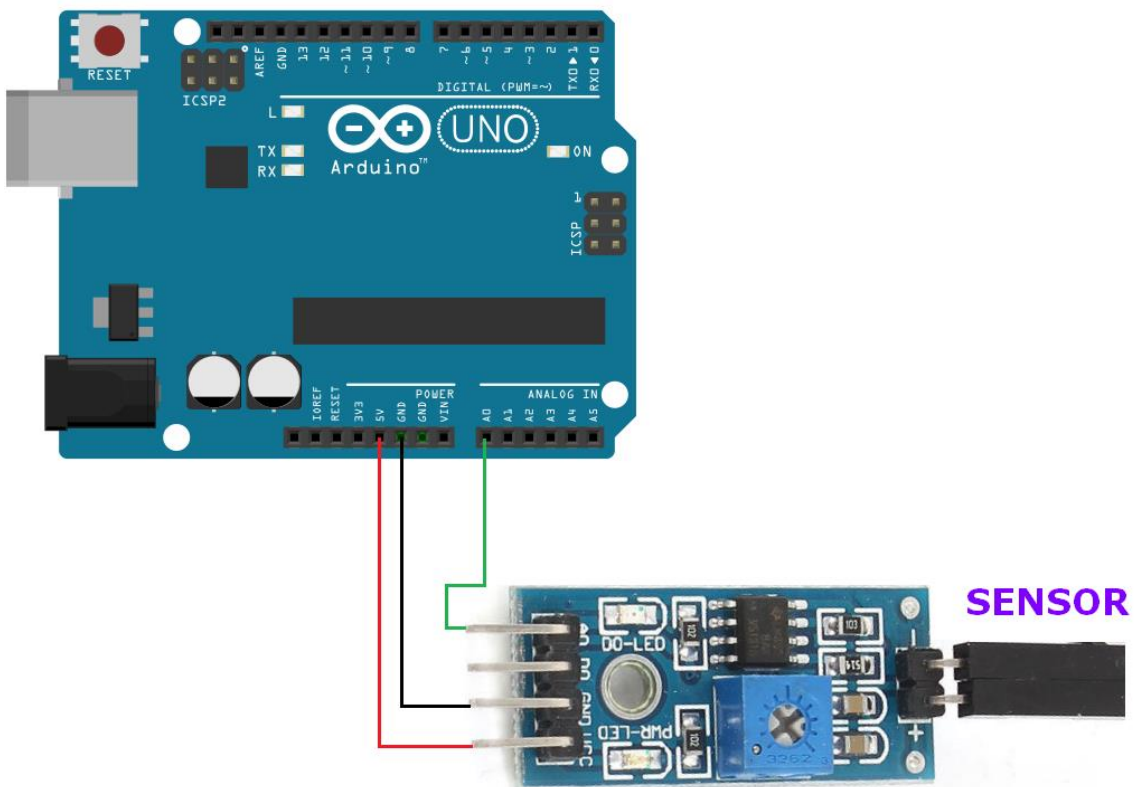
#### 3.1. De relais



### 3.2. De temperatuur sensor



### 3.3. Vochtigheidssensor



## Bibliografie

/. (2010, 10 20). *Programmeren met Arduino*. Opgehaald van Arduino: <https://oscarromero-arduino.weebly.com/programmeren.html>

Veen, P. v. (2018, nv nv). *Arduino voor gevorde en beginners*. Opgehaald van Arduino cursus 2018: nv