Internet of Things Workshop

Solderen en Programmeren

Ruben Smit en Hilke van den Born

Inhoudsopgave

1. IoT Workshop - Solderen en programmeren

Welkom bij de Internet of Things Workshop van C.S.V. Alpha. In deze handleiding zul je alle informatie vinden die nodig is om de ledring in elkaar te zetten. Ook zijn er een aantal voorbeeld projecten toegevoegd waarmee je kunt leren hoe je microcontrollers kunt programmeren.

Deze handleiding bestaat uit drie delen. Allereerst een beschrijving hoe de hardware in elkaar gezet moet worden. Vervolgens hoe deze in de behuizing geplaatst moet worden. Daarna wordt beschreven hoe je code naar de microcontroller kunt uploaden en hoe je de verschillende onderdelen van de microcontroller programmeerd.

1.1 Benodigdheden

Voor deze workshop zijn een aantal benodigdheden. Het gros hiervan wordt bij de workshop mee geleverd. Het is mogelijk om ook een telefoonlader en/of soldeerbout te bestellen mocht je deze niet hebben. Natuurlijk kun je ook altijd de soldeerbout van iemand anders lenen.

1.1.1 IoT workshop kit

Artikel	Aantal
WEMOS D1 Mini	1
10K Ohm potentiometer	1
Push button	1
1M Ohm weerstand	1
WS2812 60 pixel led ring	1
Strip header pins	1
Draad Rood	15 cm
Draad Zwart	15 cm
Draad Geel/Paars	15 cm
Frisbee Wit	1
Soldeertin	1 strip

In onderstaande afbeelding zijn alle onderdelen van de kit behalve het soldertin te zien.

1.1.2 Zelf aanleveren

Artikel	Aantal
5V 1,5A USB voeding (telefoonlader)	1
Micro USB kabel	1
Vel A4 papier	1

1.1.3 Gereedschap

Artikel
Soldeerbout
Schaar
Tape
Lijm
Boor (of ander gereedschap om een gat in een frisbee te maken)
Computer of laptop
Kabel striptang of multitang

1.2 Wat is IoT?

Welkom in 2021! Toen in 1988 Pet Beertema als eerste Nederlanders toegang kreeg tot het NSFnet, later het internet genoemd, had niemand ooit kunnen bedenken dat ooit bijna alle apparaten verbonden zouden zijn met het internet.

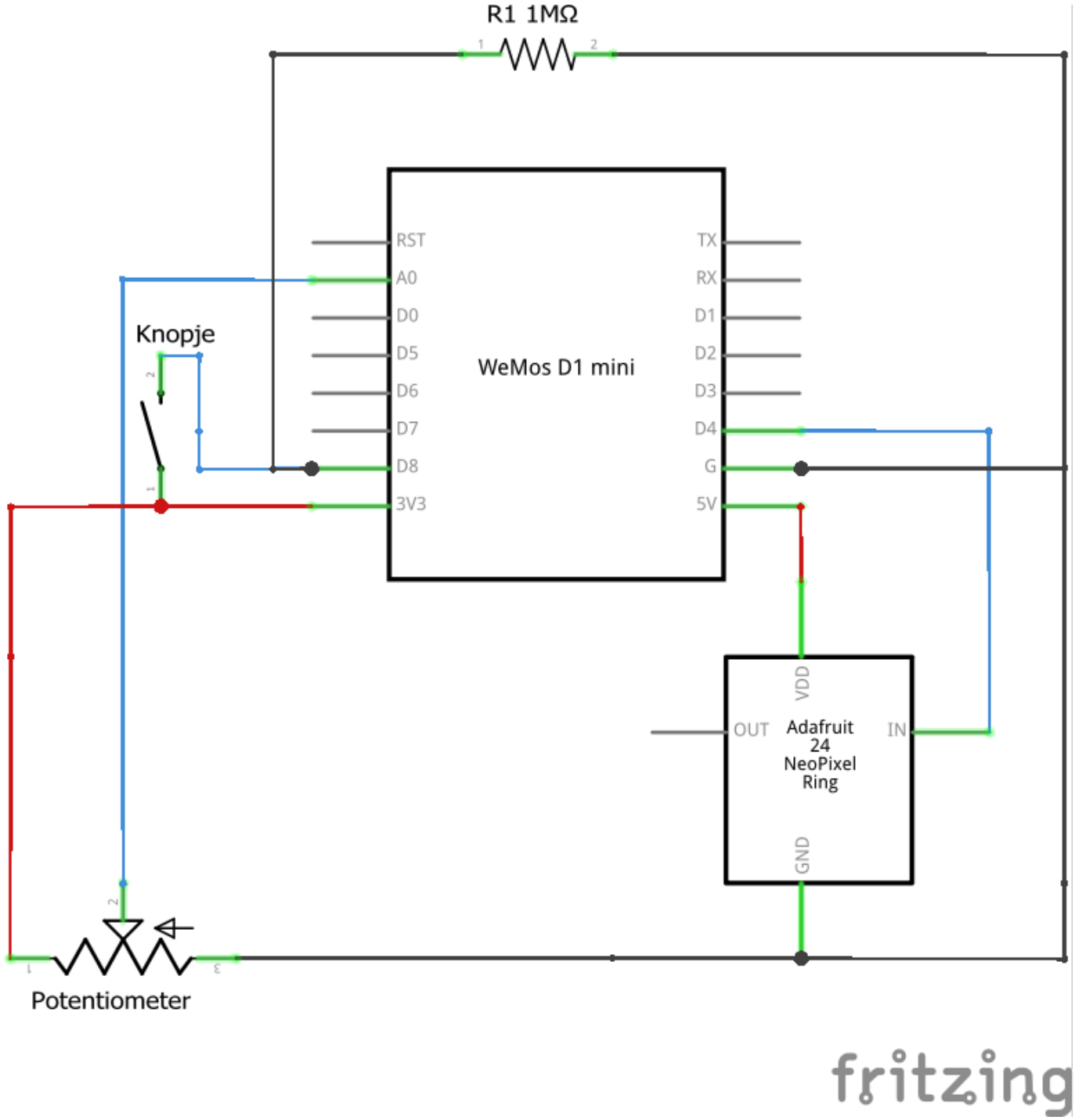
In 2020 zijn er over de gehele wereld zo'n 26 miljard apparaten verbonden aan het internet. Denk hierbij aan mobieltjes, laptops, koelkasten, lampen, tandenborstels en zelfs vuilnisbakken! Al deze apparaten verzamelen informatie en versturen die naar het internet, naar elkaar of naar een cloud. Deze heuse 'communicatie' en het versturen van informatie wordt the Internet of Things genoemd. Alles is met elkaar verbonden en bijna alle apparaten kunnen input vanuit het internet of vanuit andere apparaten halen.

Ik hoor je al denken: 'Nou, spannend dit maar wat heb ik hier aan dan?' Hartstikke veel! Alle informatie die binnen komt via apparaten kan worden verwerkt en dat kan weer gebruikt worden om smart cities aan te leggen of de landbouw te verbeteren.

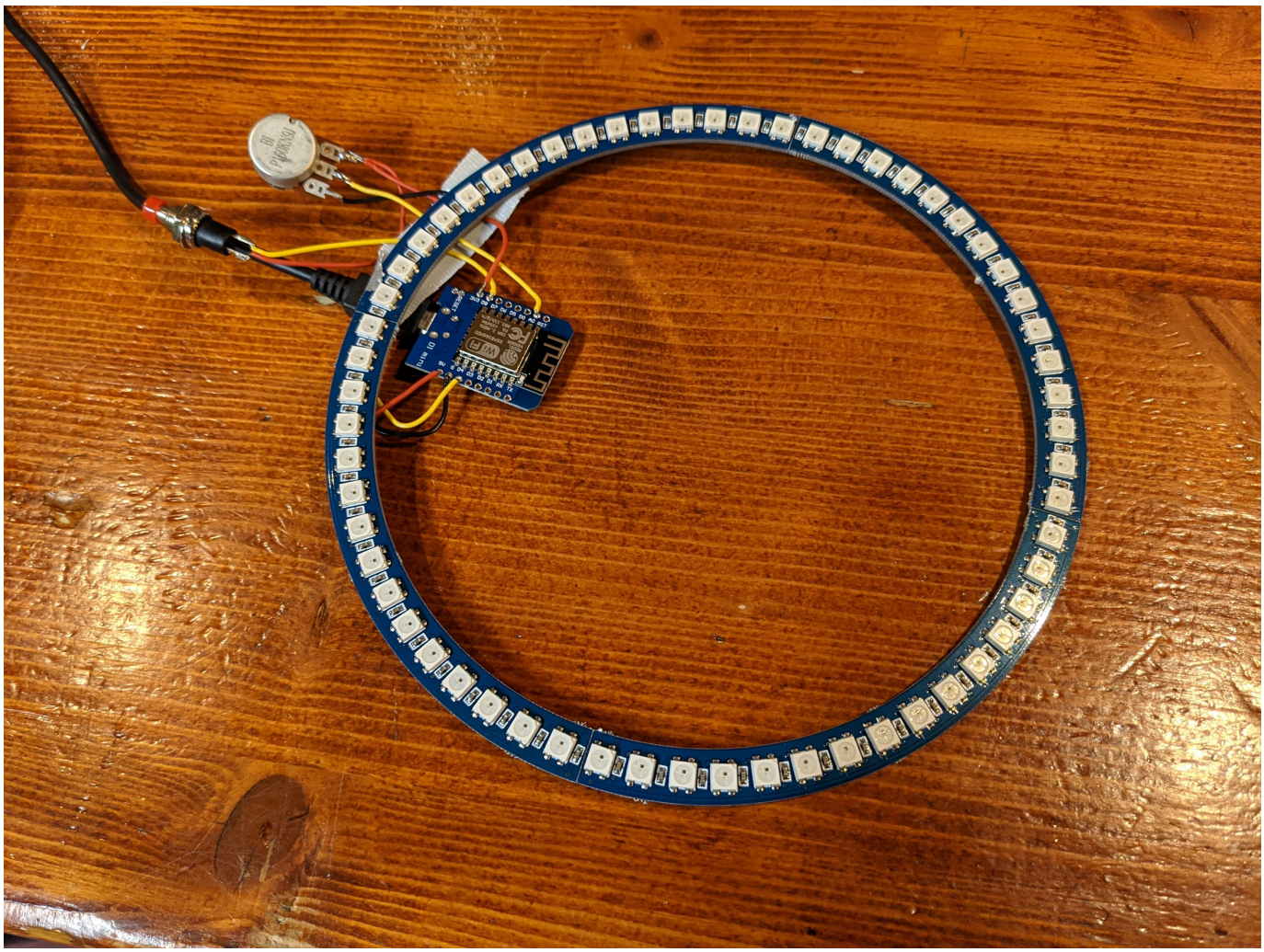
2. Hardware

2.1 Hardware overzicht

Tijdens deze IoT workshop gaan we een IoT lamp maken. dit is het schema voor het te solderen circuit:



En als het solderen gelukt is dan moet het eindresultaat er ongeveer zo uit komen te zien!



2.1.1 Draden voorbereiden

Als het goed heb zitten er 3 draden van 15cm in de kit. Knip deze draden in stukken van 5 cm en strip de isolatie van de uiteinden zodat deze gesoldeerd kunnen worden.

2.2 Solderen

Je kunt dit hoofdstuk overslaan als je alleen maar wilt bouwen

Solderen is een techniek die iedereen die wat met elektronica doet zou moeten beheersen. Door middel van solderen kun je twee elektronische onderdelen met elkaar verbinden en ontstaat er een stevige (elektronische) verbinding. We zullen kort uitleggen welk gereedschap hiervoor nodig is en hoe je verschillende onderdele soldeert. Ook zullen we uitleggen hoe je fouten kunt corrigeren door middel van desolderen.

2.2.1 Soldeergereedschap

Je hebt niet heel veel gereedschap nodig om te solderen. Hieronder beschrijven we de basis benodigdheden.

Soldeerbout

De soldeerbout gebruik je om het soldeer mee te verhitten zodat het aan de elektische componenten gesmolten kan worden. Ze komen in twee varianten, pen en pistool vorm. Voor beginners is de pen vorm het handigst. Probeer er een te kopen waarvan de temperatuur instelbaar is en met een wattage tussen de 15W en 30W. De meeste soldeerbouten hebben een verwisselbare punt. Er zijn twee gangbare vormen hierin, de conische en de platte punt. Conische punten worden vooral gebruikt voor het precies solderen van kleine componenten, platte punten voor grotere componenten en draden. Voor beginners is een conische punt aan te raden.

Een soldeerbout kan extreem heet worden!

Ja duh! Maar goed, we willen toch even waarschuwen want als je niet oppast kun je je akelig verbranden of brand veroorzaken. Doe voorzichtig. Tot zover ons verantwoordelijkheidsgevoel.

Spons

Het is belangrijk om de punt van je soldeerbout schoon en vrij van oxidatie te houden. Daarvoor kan een spons gebruikt worden. Punten met oxidatie worden zwart en accepteren geen soldeer meer. Je kunt hiervoor een vochtige conventionele spons gebruiken, maar dit verkort wel de levensduur van je punt. Een alternatief is gebruik te maken van een messing spons.

Gebruik geen schuurspons maar een cellulose spons

Schuursponzen zijn gemaakt van plastic en dat smelt bij de hoge temperaturen van de soldeerbout. Gebruik in plaats daarvan een natuurspons of een speciale cellulose soldeerbout spons.

Soldeer

Soldeer is een metaallegering met een laag smeldpunt. Het bestaat in zowel loodhoudende als loodvrije variaties in draadvorm. In de kern van de draad zit een materiaal genaamd flux, dit zorgt voor een betere soldeerverbinding. De meest gebruikte variant soldeer is loodvrijde soldeer met harskern. De loodhoudende variant is minder populair vanwege gezondheidsrisico's.

De optimale soldeertemperatuur bij loodhoudende soldeer is rond de 330°C. Voor loodvrije soldeer is dit 350°C.

Let op bij het gebruik van loodhoudende soldeer

Zorg voor goede ventilatie en was je handen naderhand. De dampen kunnen schadelijk zijn voor je gezondheid.

Overig

Een houder voor je soldeerbout is aan te raden als je vaker soldeert, daarmee voorkom je dat je brandt aan de punt of iets in de fik zet. Een ander handig hulpmiddel bij het solderen is een "helpend handje". Met de klemmetjes kun je twee componenten bij elkaar houden zodat je je handen vrij hebt om te solderen. Een goedkoop alternatief hiervoor is houten knijpers en ijzerdraad.

2.2.2 Veiligheid

Zorg er voor dat je soldeert in een goed geventileerde ruimte. Bij het solderen komen schadelijke dampen vrij die slecht zijn voor je ogen en je longen. Was achteraf altijd je handen. Zorg voor een ondergrond die hittebestendig is en gebruik hulpmiddelen zoals tangetjes om de materialen die je aan elkaar soldeert vast te houden.

2.2.3 Soldeerpunt vertinnen

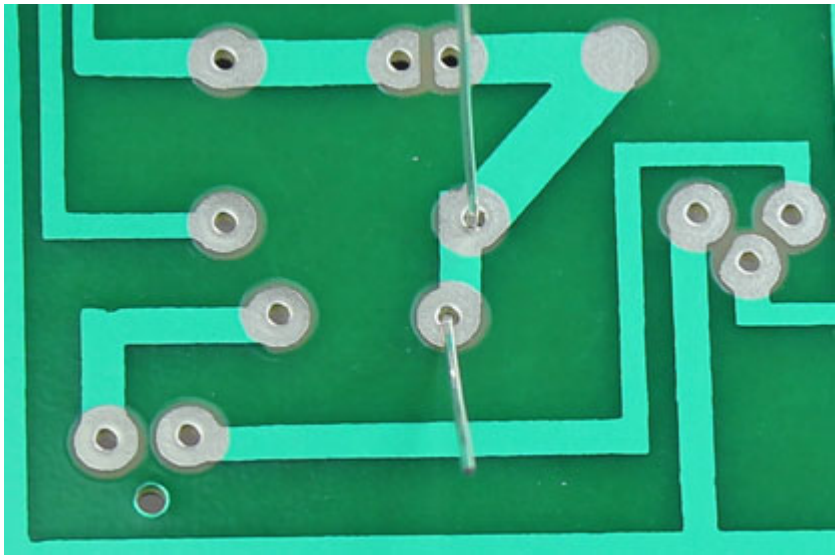
Voordat je begint met solderen en nadat je klaar bent met het solderen is het slim om de punt van je soldeerbout te vertinnen. Hiermee verbeter je de hitteoverdracht van je soldeerbout en gaat het solderen een stuk makkelijker. Daarnaast voorkomt het slijtage aan de punt en ontstaat er minder oxidatie.

1. Controleer of de punt goed vastgeschroeft zit voor een optimale hitteoverdracht.
2. Zet je soldeerbout aan en stel hem, indien mogelijk in de juiste temperatuur in voor je soldeer.
3. Maak, wanneer de soldeerbout heet is, de punt van je soldeerbout schoon met een vochtige spons of een messing spons.
4. Coat de punt van je soldeerbout in een dun laagje soldeer.

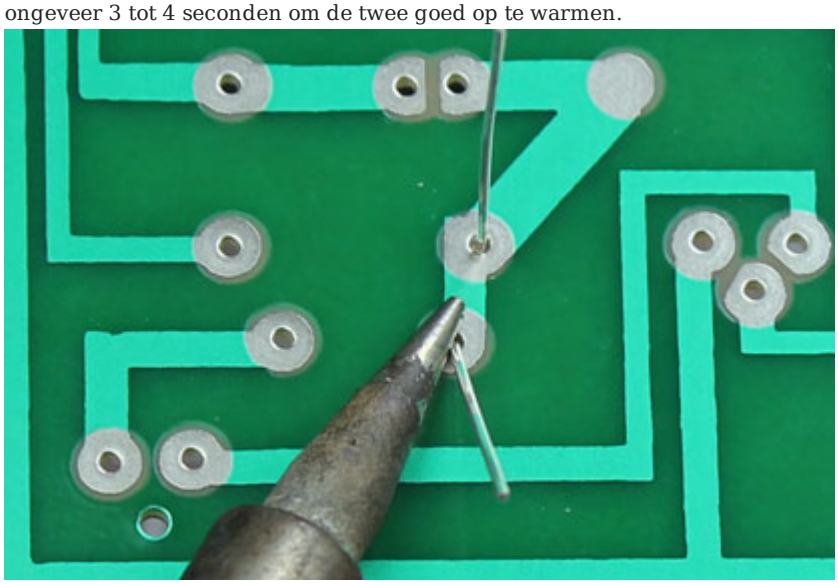
2.2.4 Componenten solderen

Je soldeert als volgt een component aan een printplaat.

1. Steek de draad door het gat in de printplaat en buig hem ongeveer 45 graden. Op deze manier blijft de draad goed zitten en krijg je een betere verbinding bij het solderen.

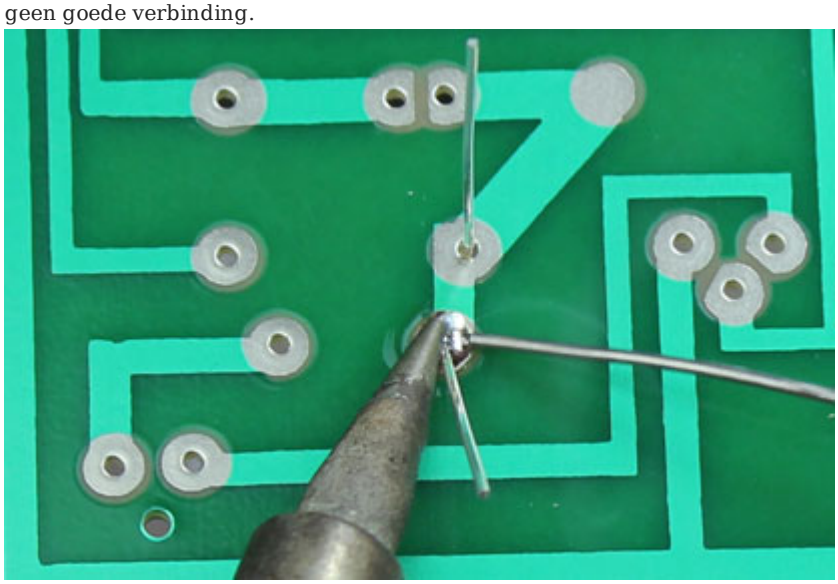


2. Verhit de verbinding. Dit doe je door met de soldeerpunt zowel het koper op de printplaat en de draad aan te raken. Doe dit voor ongeveer 3 tot 4 seconden om de twee goed op te warmen.

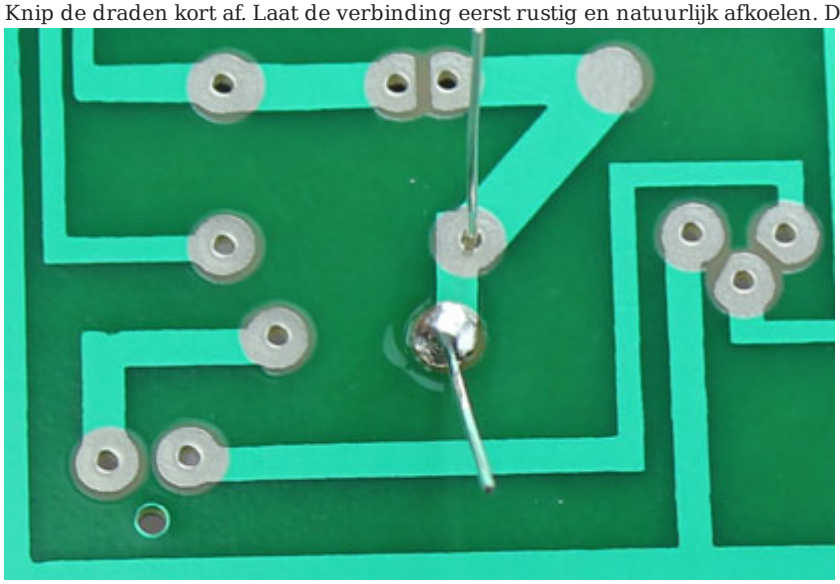


3. Voeg soldeer toe aan de verbinding. Terwijl de de soldeerpunt op zijn plek houdt voeg je soldeer toe. `Let op` raak met de soldeer niet de soldeerpunt aan maar de verbinding zelf. De verbinding moet heet genoeg zijn om de soldeer te smelten, anders krijg je

geen goede verbinding.



4. Knip de draden kort af. Laat de verbinding eerst rustig en natuurlijk afkoelen. Dit zorgt voor de meest stevige verbinding.



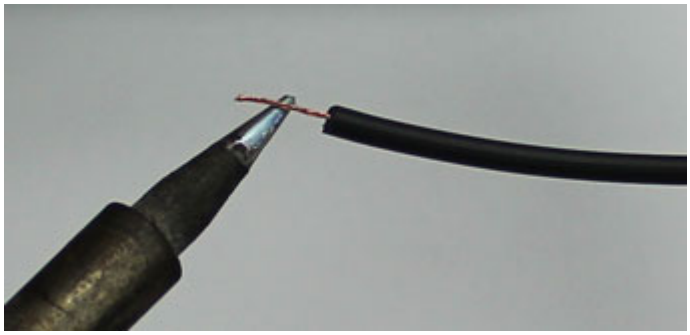
Een goede soldeerverbinding is glad en glanzend en lijkt op een vulkaan of kegelvorm. Er is net genoeg soldeer om de hele verbinding te bedekken maar niet zoveel dat het een balletje vormt.

Wil je dit graag in een mooie video uitgelegd krijgen, kijk dan deze youtube video: https://www.youtube.com/watch?v=kTURB6QboNY.

2.2.5 Draden solderen

Zo soldeer je twee draden aan elkaar.

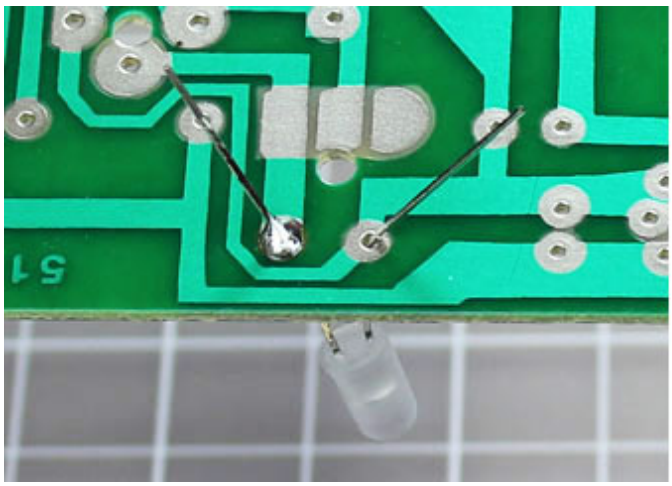
1. Verwijder de isolatie van beide draadeinden. Als het een gevlochten draad is draai je de draden in elkaar in je vingers.

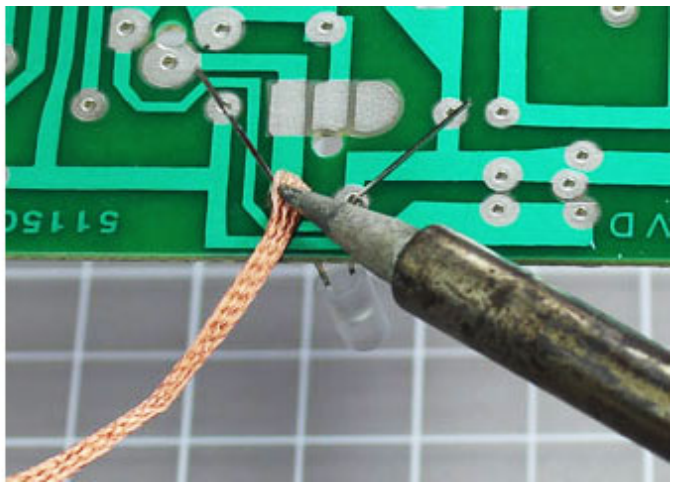
2. Met een hete soldeerbout raak je de punt van het draad aan zodat hij in 3 tot 4 seconden goed opwarmt.

3. Vervolgens raak je de draad aan met het soldeer. Hiermee leg je een dunne coating van soldeer aan om de draad. Doe dit ook voor de andere draad. Zorg ervoor dat je niet de punt van je soldeerbout aanraakt met de soldeer.

4. Houd de twee draden tegen elkaar en raak ze aan met de punt van je soldeerbout. Hiermee zouden de beide gecoate draden aan elkaar moeten smelten.


2.2.6 Desolderen

Het mooie aan solderen is dat je fouten kunt herstellen. Daarvoor kun je een verbinding desolderen met een desoldeerlont (ookwel desolderbraid of desolderwick). Deze kan overtollig soldeer "opzuigen".





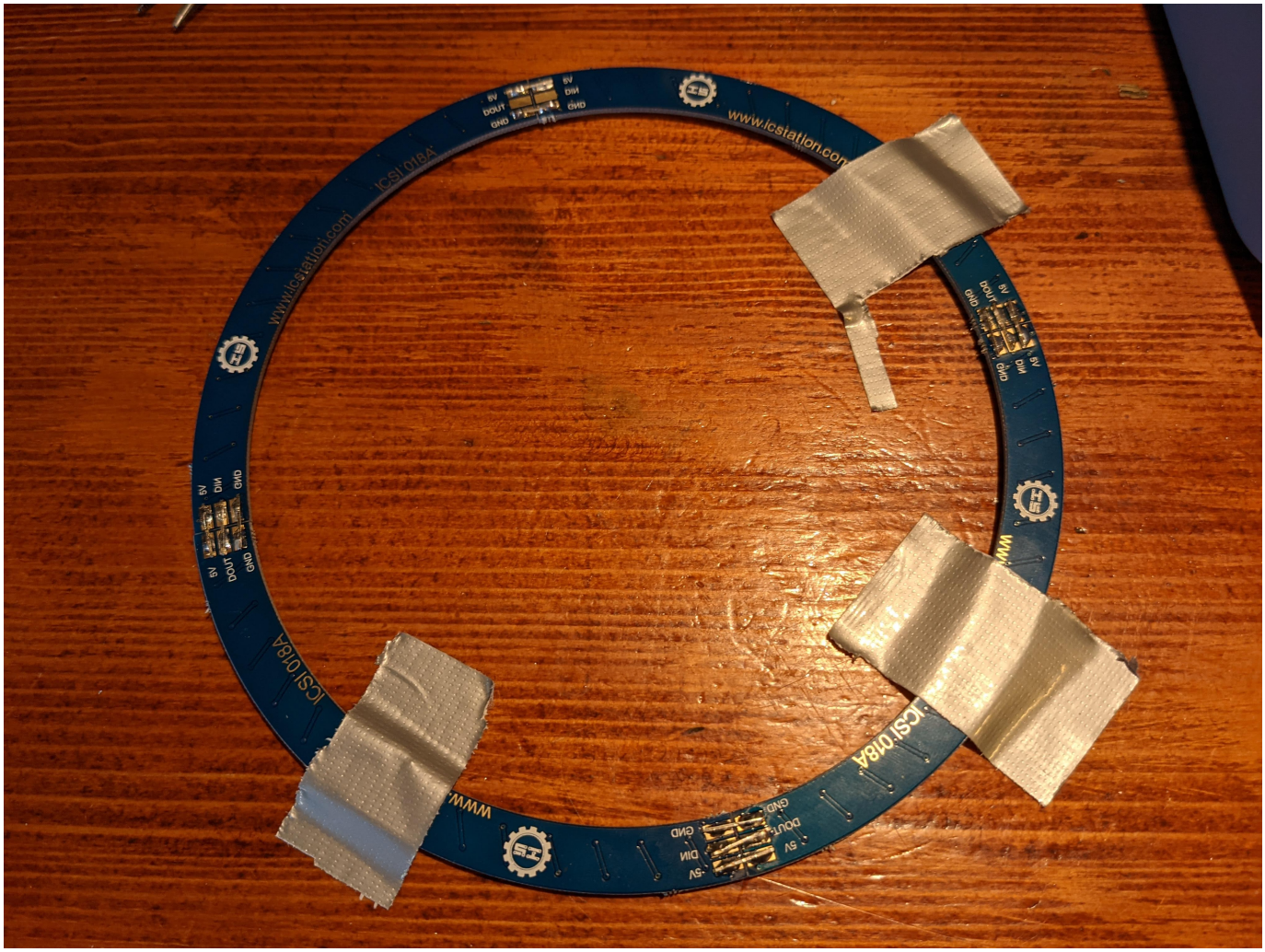
Plaats de lont op de verbinding die je wilt desolderen. Vervolgens verhit je de lont met je soldeerbout. Als het goed is wordt de soldeer in de lont opgenomen.

2.3 Ledring

We beginnen met de leukste stap, het solderen van de ledring!

2.3.1 Ring solderen

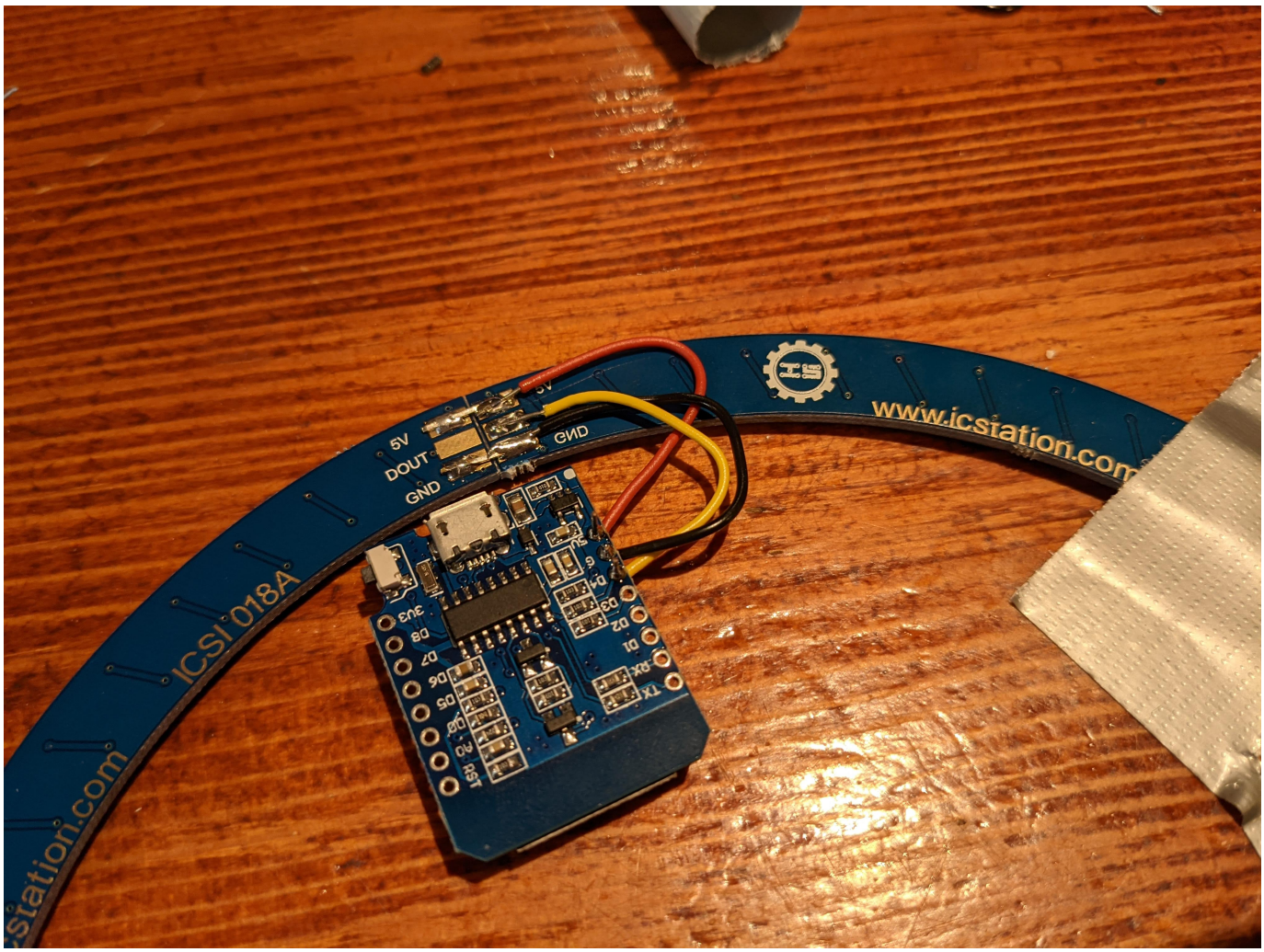
We raden aan om de ledring op tafel vast te plakken voordat je begint met solderen. Je hebt als het goed is een strip met metalen pinnetjes gekregen. Die kun je uit de strip trekken en op de ring solderen. Verbind alle `GND` en `5V` vlakken met elkaar. Verbind ook de `DIN` en `DOUT` vlakken met elkaar voor de data verbinding **op één na**. Dit zal straks het begin en einde van de ledring zijn. Het solderen van de ledring is best een priegel werkje, in deze video kun je wat tips zien hoe je dit het beste doet.



In deze closeup kun je de verbindingen zien. Hier zie je het begin en einde van de ledring.

2.3.2 Ring aansluiten

Vervolgens moet de ledring aangesloten worden op de ESP. Soldeer daarvoor draden tussen de `GND` pin van de ESP en het `GND` vlak van de ledring, de `5V` pin van de ESP en het `5V` vlak van de ledring. Voor de dataoverdracht tussen de ledring en de ESP moet je een draad solderen tussen de `D4` pin van de ESP en het `DIN` vlak van de ledring. Let goed op dat deze draad niet per ongeluk het `DOUT` vlak raakt want dan werkt de ring niet.



2.4 Voltage divider

Je kunt dit hoofdstuk overslaan als je alleen maar wilt bouwen

We willen je graag wat achtergrondinformatie geven over het aansluiten van bepaalde componenten. Deze basis zou voldoende moeten zijn om de meeste componenten op een Arduino aan te kunnen sluiten. Oftewel, om alles te bouwen wat je maar zou willen.

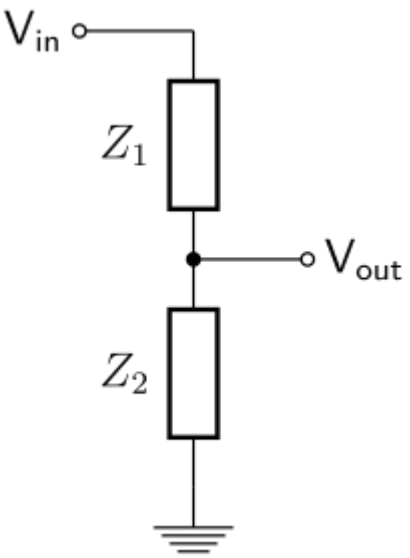
Ha wat episch dat jij dit stukje door leest! Wellicht heb je al kennis van elektronica, stroom en andere machtig mooie dingen maar het kan heel goed zijn dat je die kennis nog niet hebt! We willen je graag de voltage divider uitleggen of in het Nederlands: 'de spanningsdeler'.

Een voltage divider is een circuit die de spanning (V) die aan het begin binnen komt in kleinere deelspanningen aan het einde splitst. Dit komt door een serieschakeling van weerstanden waardoor de totale spanning over de weerstanden wordt verdeelt. Mocht je de deelspanningen bij elkaar optellen dan krijg je, ja je raad het al, de totale begin spanning. En dit is samen te vatten in de voor iedereen wel bekende formule: V=I*R

$$V_{uit} = \frac{R_{uit}}{R_{totaal}} \cdot V_{in}$$

De deelspanning die iedere weerstand krijgt hangt af van de grootte van de weerstanden en de spanning aan het einde hangt af van de verhouding tussen de twee weerstanden. Weerstanden van 9 Ohm + 10 Ohm geeft dus het zelfde resultaat als 9 kOhm + 10 kOhm.

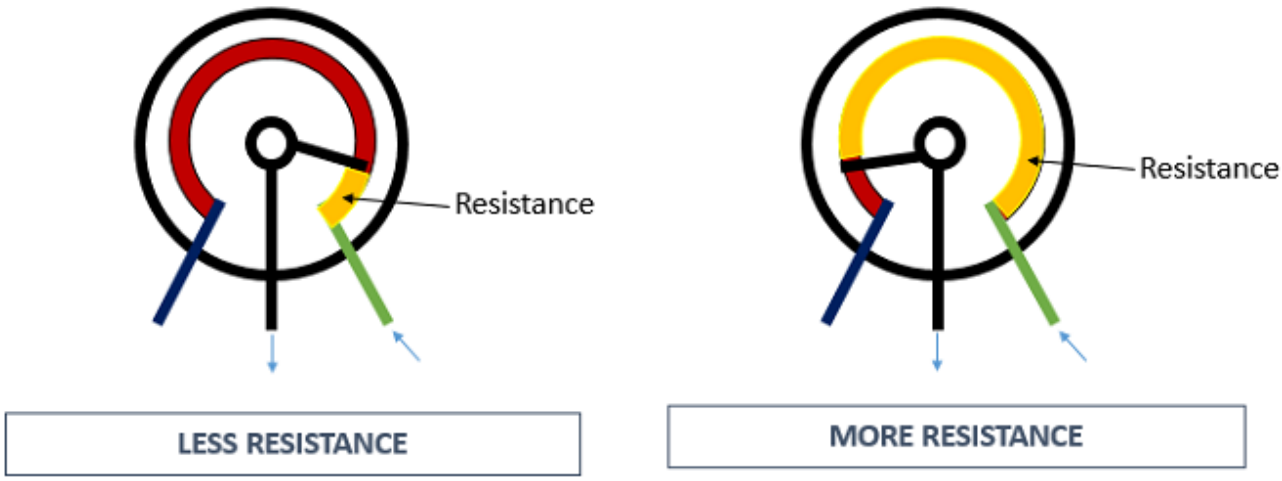
Helemaal leuk dit maar hoe werkt zo'n voltage divider dan nou precies. Zie het zo, als je begint 12V en je wil een output voltage (Vout) van 6V dan kan je een voltage divider gebruiken. Dit circuit splits de output voltage dan in bijvoorbeeld 2x 6V (6V naar ground en 6V als Vout). Belangrijk hier is dan een voltage divider de output voltage splitst in meerdere stukjes en dus niet verminderd. Als je de output voltage wilt verminderen dan moet je een transformator gebruiken :)



2.5 Potentiometer

We hebben het net over een voltage divider gehad. Allemaal leuk en aardig maar wat nou als ik steeds mijn output voltage wil veranderen. Ik ga niet steeds mijn weerstand combinatie veranderen. Heel begrijpelijk, gelukkig heeft het leven hier een oplossing voor: een potentiometer!

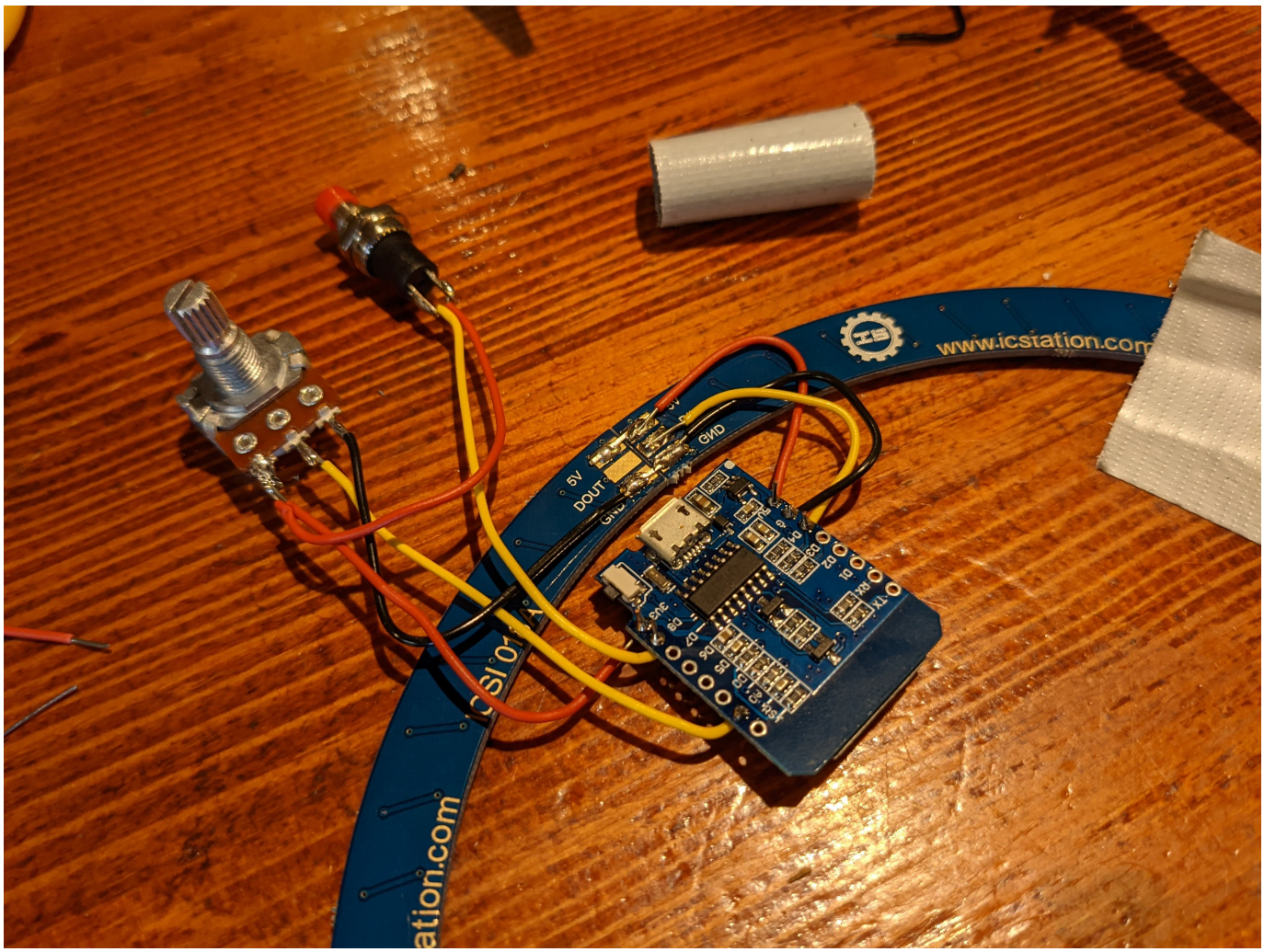
Een potentiometer is een verstelbare weestand die bestaat uit weerstandsmateriaal (dit 'remt' de stroom als het ware af) en een loper/slede. De stand van de slede kan je veranderen door aan de knop te draaien. Dit kan omdat een potentiometer drie pootjes heeft. Voor nu noemen we ze even A,B en C. De buitenste pootjes zijn A en C die zitten vast aan het weerstandmateriaal. Pootje B zit vast aan de slede.



Wanneer je dit circuit aansluit om stroom dan ga je twee van deze pootjes verbinden, dus je verbindt of pootje A en B of je verbindt B en C. Wanneer dit gebeurd moet de stroom door het weerstandsmateriaal. Hoe langer de weg van het ene pootje naar het andere pootje hoe groter de weerstand.

2.5.1 Solderen

Soldeer de middelste pin van de potentiometer met een draad aan `A0`. Een van de buitenste pinnen soldeer je met een draad aan `3V3` en de andere buitenste pin aan `GND`. Omdat de `GND` pin van de ESP al in gebruik is voor de ledring kun je de draad ook aan de `GND` van de ledring solderen.



2.6 Pull-down resistor

Je kunt dit hoofdstuk overslaan als je alleen maar wilt bouwen

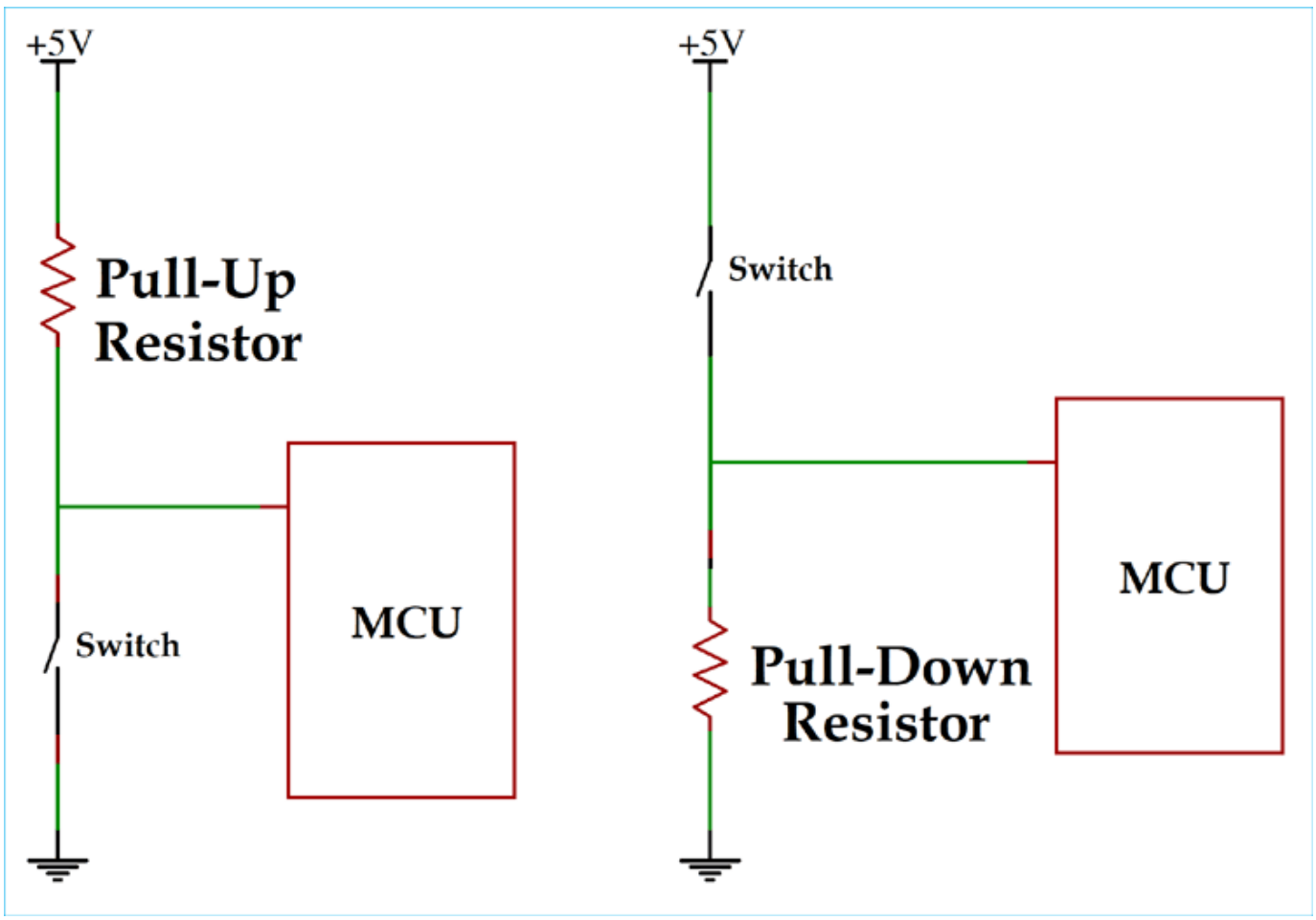
We willen je graag wat achtergrondinformatie geven over het aansluiten van bepaalde componenten. Deze basis zou voldoende moeten zijn om de meeste componenten op een Arduino aan te kunnen sluiten. Oftewel, om alles te bouwen wat je maar zou willen.

Pull-down resistors zijn weerstanden die een hele belangrijke rol hebben in veel schakelingen waarin een digitale sensor de spanning meet. Dat kan al bij een drukknopje zijn die kleine stroompulsjes van 5V stuurt.

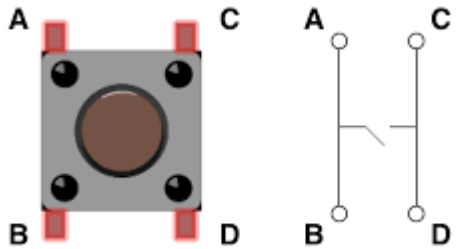
Wanneer zo'n knopje niet is ingedrukt zou er dus 0V naar een chip moeten gaan maar stiekem is dat niet zo. Vaak is er van allerlei ruis dus wel kleine stroompulsjes richting de digitale input pin. Zo'n digitale input pin meet alleen maar 0V of 5V en wanneer er ruis is kan de 'trigger drempel' van zo'n digitale pin worden bereikt waardoor die pin denkt dat er 5V doorheen gaat terwijl die eigenlijk helemaal niet zo is. Moker irritant natuurlijk.

Om dit soort dingen te voorkomen kunnen we een hele grote weerstand plaatsten tussen de digitale in pin en de ground. Dit zorgt ervoor dat de digitale in pin verbonden is met de grond en dus altijd 0V meet wanneer de knop niet is ingedrukt.

Wanneer de knop wel wordt ingedrukt wordt de digitale IN pin verbonden met 5V en zal hij dus 5V meten.



2.7 Knopje

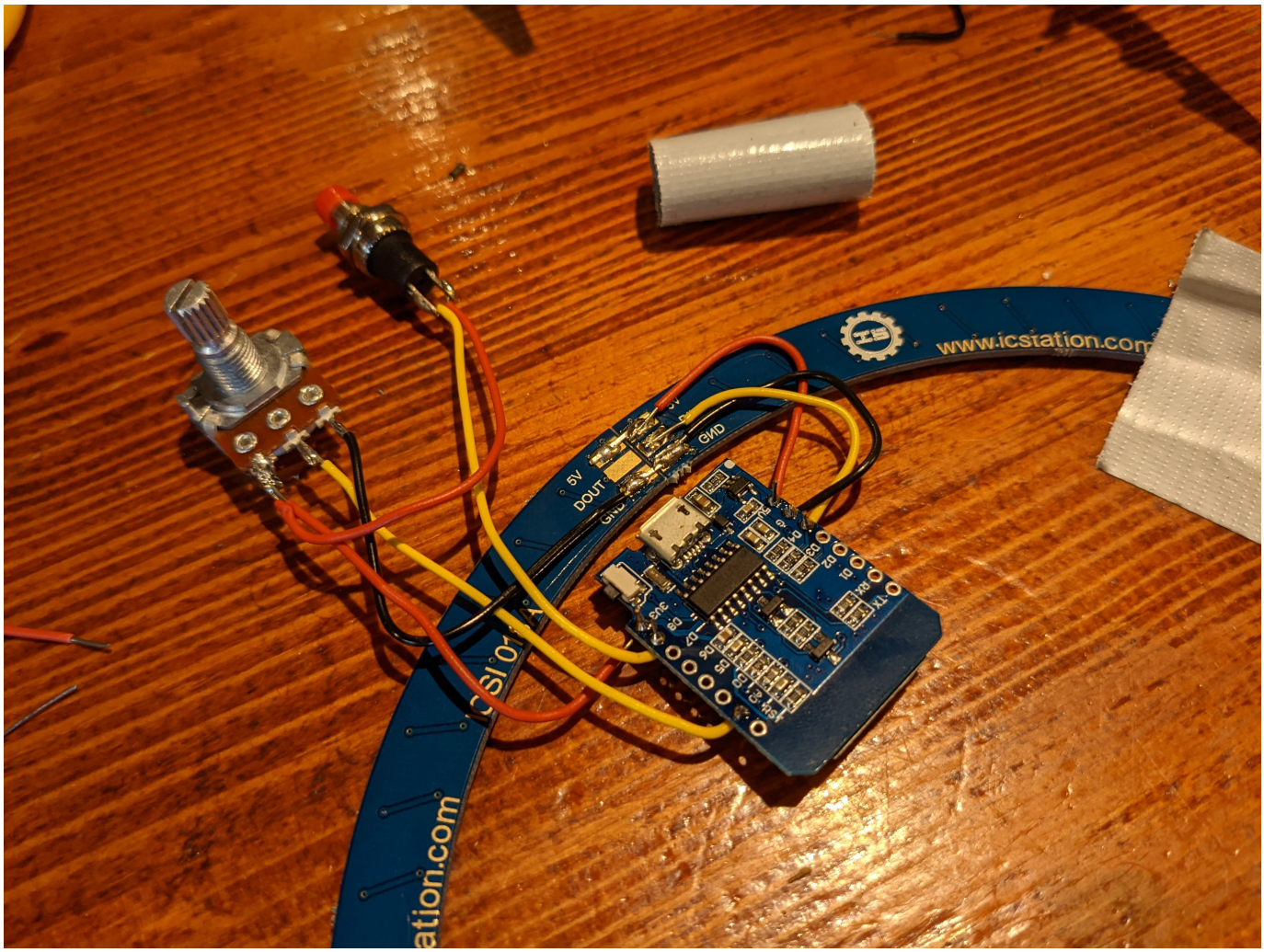


Knoppies! Helemaal leuk! Eigenlijk is een knopje net zoals een ophaalbrug. Wanneer de ophaalbrug open is kan er geen verkeer door heen en wanneer deze weer dicht is kan je er weer over heen fietsen. Zo gaat het met een knopje ook.

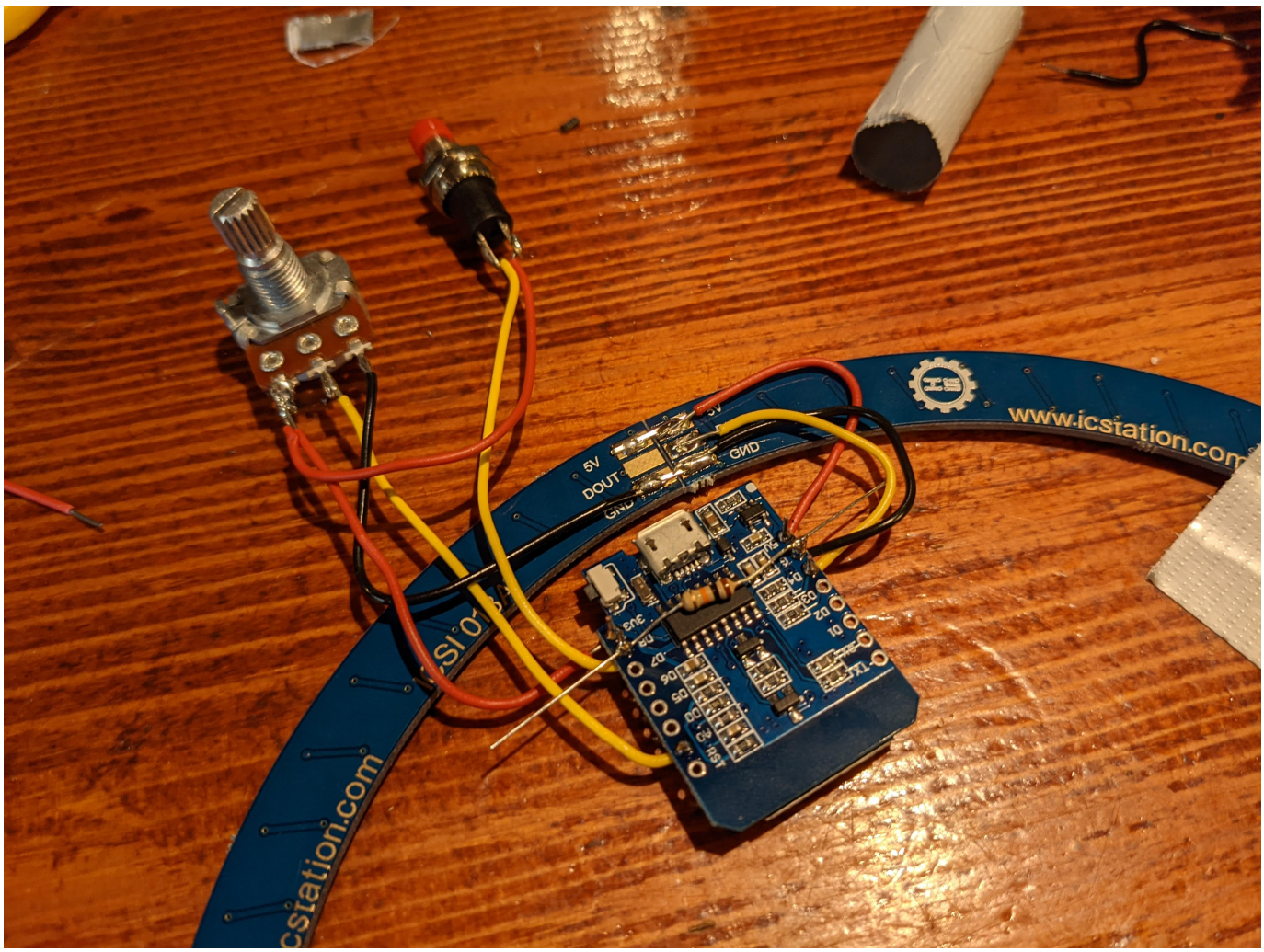
Een knopje overbrugt een gat in een circuit. Als er een gat is het circuit zit betekent dit dat de stroom niet verder kan. Wanneer een knopje dan ingedrukt wordt, wordt er als het ware een brug gecreëerd waardoor de stroom weer verder.

2.7.1 Solderen

Soldeer een draad vanaf de `3V3` aansluiting aan een van de aansluitingen van het knopje. Op de afbeelding is dit een rode draad die aan de `3V3` aansluiting van de potentiometer gesoldeerd is. Soldeer vervolgens een draad van de andere aansluiding van het knopje naar de `D8` pin van de ESP.



Vervolgens kun je de weerstand als pull-down resisor solderen. Daarvoor soldeer je de ene kant van de weerstand aan `D8` en de andere kant aan de `GND` pin. Let op dat de weerstand geen andere pinnen of elektronische componenten raakt. Je zou voor de zekerheid wat tape kunnen gebruiken.



Dit is een goed moment om je hardware te testen

Tenzij je erg veel zelfvertrouwen hebt raden we je aan om op dit moment je hardware te testen. Controleer eerst goed of geen van je draden elkaar raken waar dat niet zou moeten. Plaats zo nodig preventief wat tape. Volg de stappen in Arduino IDE om vervolgens de test code uit te voeren. Als alles correct aangesloten is zal er een regenboog over de ledring cirkelen. Door aan de draaiknop te draaien moet deze sneller of langzamer gaan draaien. Als je het knopje indrukt moet deze stoppen met draaien. Werkt alles? Gefeliciteerd! Dan kun je nu door naar de behuizing.

3. Behuizing

Helaas was het niet mogelijk om een plexiglas voorplaat te laten maken voor deze workshop. Door de pandemie en alle corona kuchschermen is plexiglas slecht leverbaar. Maar niet getreurd, alle elektronica kan in de frisbee bevestigd worden en een papiertje kan gebruikt worden als diffuuse voorkant.

3.1 Bevestigen knopje en potentiometer

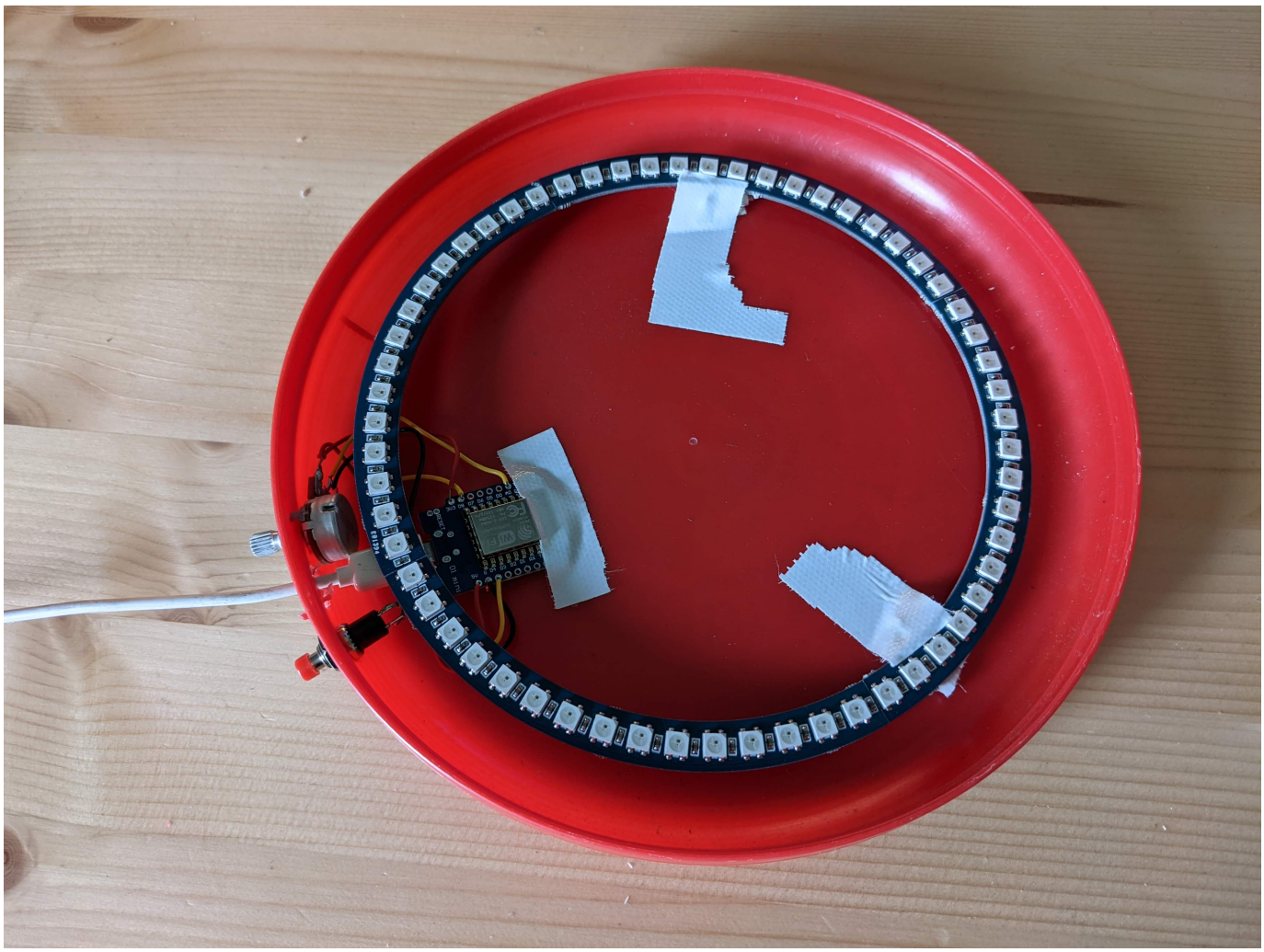
Maak drie gaten in de onderkant van de frisbee. Hier kun je het beste een houtboor voor gebruiken. Let er op dat de afstand tussen de gaten voldoende is zodat alle onderdelen naast elkaar passen.

> ⚠ **Gebruik niet je soldeerbout om de gaten te maken**
>
> Als je dat wel doet kun je de soldeerbout weggooien want dan is hij niet meer bruikbaar om mee te solderen.

Je kunt vervolgens het knopje en de potentiometer bevestigen met de meegeleverde ringetjes en moertjes. Druk vervolgens het dopje op de potentiometer.

3.2 Bevestigen elementen

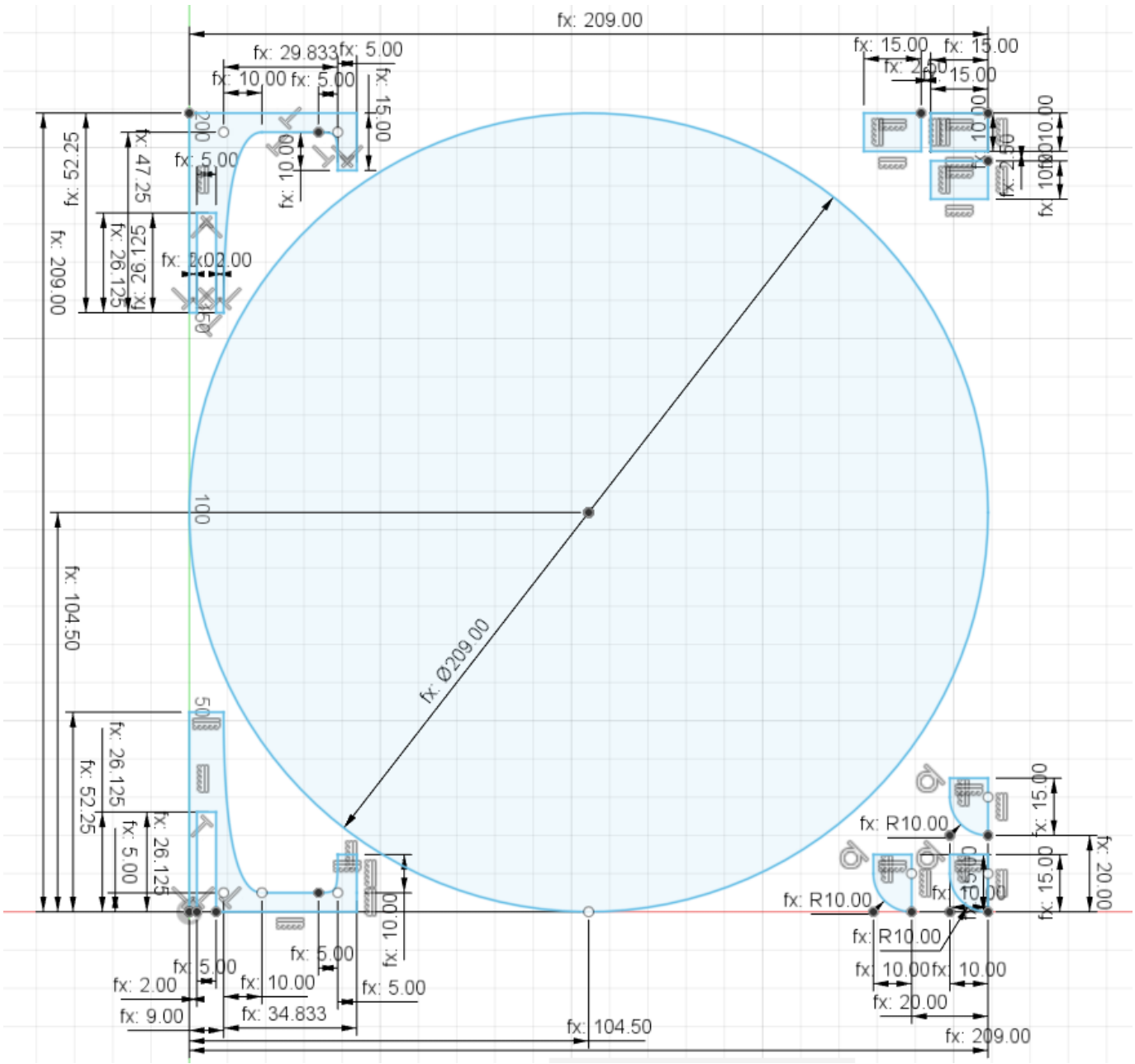
Je kunt voor nu alles in de frisbee tapen. Zorg er voor dat de hoogte van de ledring ten opzichte van de frisbee constant is. Zo is de belichting mooi uniform. Hoe verder je de ledring naar voren plaatst hoe duidelijker de individuele pixels te zien zijn.



Gebruik een vel A4 papier om het licht mooi difuus licht te krijgen. Als je wit bakpapier gebruikt kun je de pixels beter zien en komt er meer licht doorheen.

3.3 Zelf lasersnijden

Heb je zelf een lasersnijder en een plexiglas plaat van 5 milimeter dik dan kun je de behuizing zelf snijden. Download het lasersnij bestand hier. Kijk in onderstaande afbeelding voor de afmetingen.



4. Software

4.1 Software

Arduino is een opensource-computerplatform bedoeld om microcontrollers eenvoudig te maken. Dit platform is bedoeld voor iedereen die geïnteresseerd is in het maken en ontwerpen van slimme en creatieve objecten. Als basis hiervoor is de Arduino IDE waarmee aan de hand van de Arduino library voor C++ programma's geschreven kunnen worden voor de diverse Arduino microcontrollers.

Door de grote populariteit van Adruino is het ondertussen ook mogelijk om met de Arduino IDE en de bijbehorende programmeertaal vergelijkbare microcontrollers te programmeren, zoals de microcontroller die we voor dit project gebruikt hebben.

Allereerst zullen we je laten zien hoe je de Arduino IDE installeerd en code upload naar de microcontroller. Vervolgens zullen we een korte introductie geven in het programmeren in C++ en aansluitend de verschillende facetten van embedded programmeren doorlopen.

> **Liever een uitgebreide IDE gebruiken?**
>
> Het is ook mogelijk om Arduino's en andere microcontrollers te programmeren en flashen door middel van de PlatformIO plugin voor Visual Studio Code of een van de andere ondersteunde editors.

4.1.1 Arduino IDE installeren

De Arduino IDE kan gebruikt worden om code te schrijven voor en te uploaden naar microcontrollers. Deze is standaard geschikt voor alle Arduino microcontrollers en met wat aanpassingen ook voor andere microcontrollers. Download en installeer de Arduino IDE vanaf https://www.arduino.cc/en/software.

4.1.2 ESP8266 add-on

De microcontroller die wij voor dit project gebruiken is een `WEMOS D1 Mini` op basis van een `esp8266`. Dit is geen standaard Arduino product en wordt daarom ook niet standaard door de Arduino IDE ondersteund. Daarom moeten we een add-on toevoegen aan de Arduino IDE om code te kunnen compileren en uploaden naar de microcontroller. Volg daarvoor, nadat je de Arduino IDE geïnstalleerd hebt, de volgende stappen:

1. Ga in je Arduino IDE naar **File > Preferences**
2. Voeg **http://arduino.esp8266.com/stable/package_esp8266com_index.json** toe aan het **Additional Boards Manager URLs** veld en klik daarna op **Ok**.
3. Open de Boards Manager, ga naar **Tools > Board > Boards Manager...**.
4. Zoek naar **ESP8266** en druk op de installeerknop bij **ESP8266 by ESP8266 Community**.
5. Als het goed is zou de add-on binnen enkele seconden geïnstalleerd moeten zijn.

4.1.3 Adafruit Neopixel library installeren

Om alle leds op de ledring aan te sturen is een library vereist. Een library is een collectie code die het makkelijker maakt om bepaalde sensoren, displays, modules en in ons geval een ledring te verbinden met een microcontroller. Er zijn vele verschillende libraries beschikbaar om te installeren voor diverse toepassingen.

Volg de volgende stappen om de Adafruit Neopixel library te installeren:

1. Ga naar **Sketch > Include Library > Manage Libraries...**.
2. Zoek naar **Neopixel** en druk op de installeerknop bij **Adafruit NeoPixel**.
3. Als het goed is zou de library binnen enkele seconden geïnstalleerd moeten zijn.

> **Let op! Installeer niet de Adafruit DMA neopixel library**
>
> Dat is een andere library, dus dat gaat niet werken met onze ledring.

Als je gebruik wilt maken van een library in je programma kun je een eerder geïnstalleerde library toevoegen door te gaan naar **Sketch > Include Library** en hem daar te selecteren.

4.1.4 Code compileren en uploaden

Wanneer je tevreden bent met je code is het tijd om deze te compileren en te uploaden naar de microcontroller die je met een USB kabel aan je computer hebt verbonden.

Om je code te compileren druk je op het groene vinkje linksboven. Bij de compilatie wordt gecontroleerd of er geen fouten in je code zitten en wordt de code omgezet naar instructies die voor de microcontroller te begrijpen zijn. Als er nog fouten in de code zitten dan zal dat in de console onder aan de IDE weergeven worden.

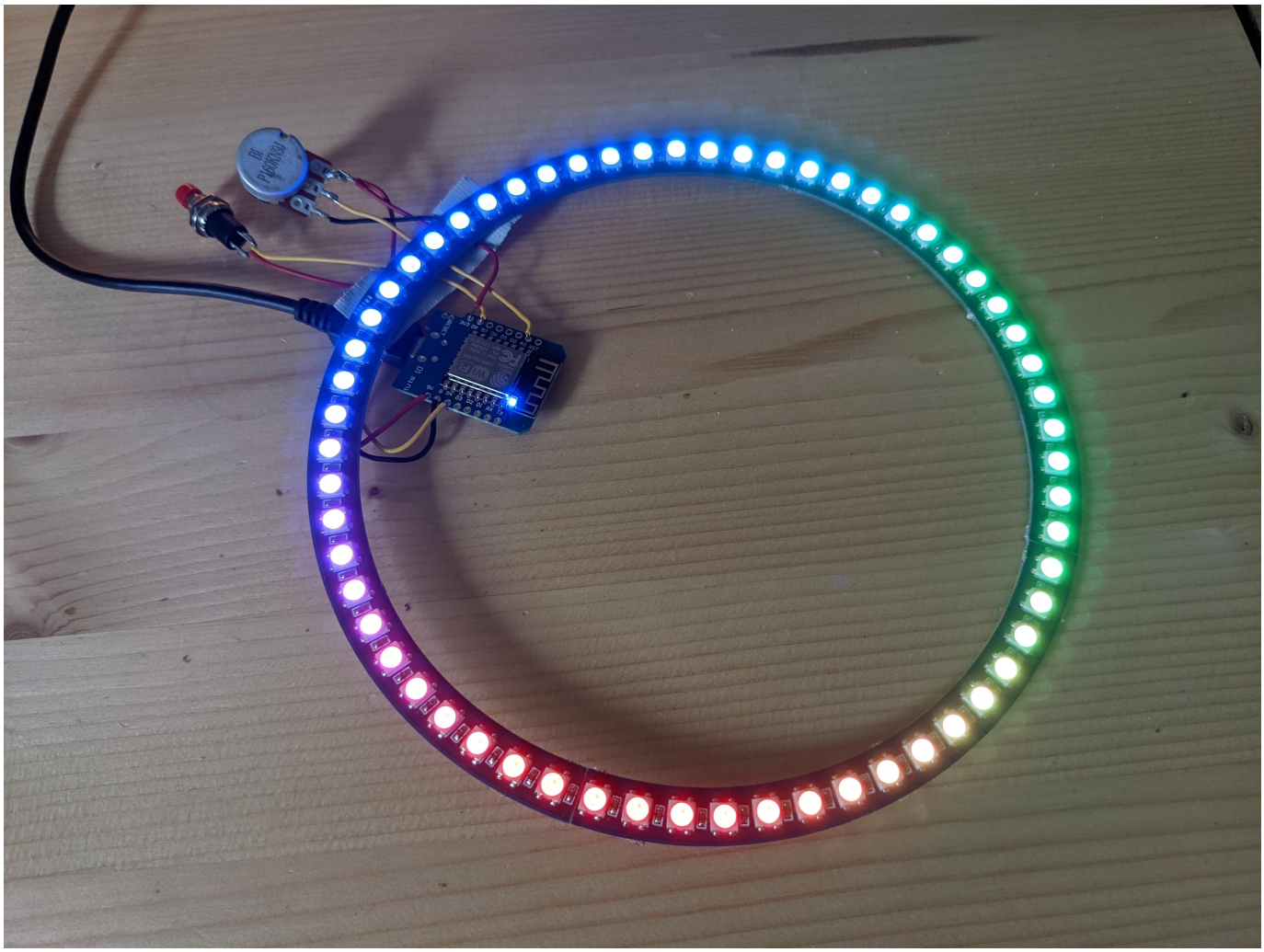
Is je code gecompileerd dan kan deze geuploaded worden naar de microcontroller. Volg daarvoor de volgende stappen:

1. Selecteer het juiste board, ga naar **Tools > Board** en kies uit de lijst de **LOLIN(WEMOS) D1 R2 & mini**.
2. Selecteer de juiste COM poort onder **Tools > Port**.
3. Druk op de upload knop, dit is het groene pijltje links boven naast het vinkje.

> **Welke COM poort moet ik gebruiken? Ik zie er meerdere.**
>
> Trek de USB kabel van de ledring uit je computer, en onthoud welke COM poorten er in het lijstje staan. Stop de USB kabel er weer in en kijk welke er bij gekomen is. Dit is de COM poort van de ledring. Zie je geen verschil? Gebruik dan een andere USB poort op je computer.

Als alles goed gaat zal in de console onder in de IDE weergeven worden dat het uploaden is gelukt.

Test code



Gebruik de volgende code om te testen of je hardware goed werkt. Als alles correct aangesloten is zal er een regenboog over de ledring cirkelen. Door aan de draaiknop te draaien moet deze sneller of langzamer gaan draaien. Als je het knopje indrukt moet deze stoppen met draaien.

Help! Er is wat mis!

Gedraagt je hardware zich niet zoals hierboven beschreven? Blijf kalm en kijk in de probleemoplosser.

```
#include <Arduino.h>
#include <Adafruit_NeoPixel.h>

#define POT_PIN A0
#define BTN_PIN D8
#define LED_PIN D4
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);
long firstPixelHue = 0;

void rainbow(int wait) {
  if (firstPixelHue < 5 * 65536) {
    firstPixelHue += 256;
    for (int i = 0; i < strip.numPixels(); i++) {
      int pixelHue = firstPixelHue + (i * 65536L / strip.numPixels());
      strip.setPixelColor(i, strip.gamma32(strip.ColorHSV(pixelHue)));
    }
    strip.show();
    delay(wait);
  }
  else {
    firstPixelHue = 0;
  }
}

void setup() {
  strip.begin();
  strip.show();
  strip.setBrightness(100);

  pinMode(BTN_PIN, INPUT);
}

void loop() {
  if (!digitalRead(BTN_PIN)) {
    rainbow(analogRead(POT_PIN)/100);
  }
}
```

4.2 Introductie

In dit hoofdstuk hopen we je een basis uitleg van programmeren in C++ te geven met de Arduino library. We zullen alle belangrijke punten en concepten kort aanstippen. Hopelijk is dit voldoende om mee aan de slag te gaan of heb je hiermee voldoende aanknopingspunten om verdere informatie op te zoeken.

4.2.1 Basis programma

Ieder Arduino script heeft altijd twee blokken: void setup() en de void loop().

```
#include <Arduino.h>

void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

void setup()

De void setup() staat aan het begin van het script. In de void setup() staat alles wat we nodig gaan hebben tijdens het runnen van het script. Je kan het zien als een soort van boodschappenlijst alleen vervangen we de meel en boter voor pins en variablen. Zoals je wellicht heb gezien staat er na de void setup() twee accolades {}. Tussen deze twee accolades wordt alles gezet wat we in de setup willen hebben.

De void setup() wordt maar 1 keer gelezen door de Arduino en dat is na het opstarten van het programma.

void loop()

In de void loop() vertel je de arduino wat het moet doen. Het is een beetje hetzelfde als het recept van een taart. We hebben tijdens de void setup() eerst boodchappen gedaan en nu gaan we met de void loop() het recept volgen. De instructies in de void loop() vorden oneindig lang herhaald, vandaar ook de naam!

Net zoals bij de void setup() begint en eindigd de void loop() met accolades. Het grote verschil, naast de herhaling, is dat er in de void loop() meerdere blokjes code kunen staan. Deze blokjes code noem je methods en die hebben ook allemaal weer hun eigen accolades. Zo verdeel de code in stukje met ieder een eigen functie.

Puntkomma

Iets waar arduino sketches heel erg van houden zijn puntkommas. Achter iedere regel moet een puntkomma! Met een ; sluit je een regel af en zeg je als het ware tegen de arduino: 'goed gedaan jonge, nu aan het werk met de volgende regel'. Het vergeten van een ; is ook een van de meest voorkomende redenen waarom een script het niet doet.

4.2.2 Comments

Naast het feit dat je in een script tegen de Arduino vertelt wat het moet doen is het soms ook even heel fijn om tegen jezelf te praten of om even uit te leggen wat je aan het doen bent. Voor deze gevallen kan je een comment plaatsen.

Wanneer je een comment plaatst zeg je eigenlijk tegen de Arduino:'Wat ik hier opschrijf gaat jou helemaal niets aan en daar hoef jij dan ook niets mee te doen'.

```
// single line comment
```

```
/*
 Multi
 line
 comment
*/
```

4.2.3 Variabelen

Goed, het is tijd om een jeugdtraumaatje op te halen. Ik neme je mee terug naar de rekenles op de basisschool. Je zit voor in de klas, let goed op en opeens vraagt de juf of meester aan jou: 'Als Pietje 9 appels heeft en hij geeft 4 appels weg, hoeveel appels heeft Pietje dan nog?' Met de volste overtuiging zeg jij: '4 juf'. Wat heb je lekker opgelet en je kan niet wachten op het complimentje dat je van de juf gaat krijgen. Maar tot je grote schrik krijg je helemaal geen comlimentje maar de juf zegt: '4 wat? 4 koeien? 4 broden? 4 euro?' Oh ja! Dat wa ook zo: '4 appels juf!'

Net zoals de 4 appels moet ook alles in een script een naam hebben. Dit noemen we variablem. Variablen vertegenvoordigen een getal of een tekst met een logische naam. Voorbeelden zijn bijvoorbeeld ledPin of onTime. Variablen kun je waardes geven.

Even weer terug naar het jeugdtrauma. Appels is in dat geval het variable. Aan het begin (void setup()) had Pietje 9 appels en tijdens de eerste run van de void loop() heeft pietje er 4 weg gegeven. Hier kan je goed zien dat de waardes van variablen door het hele script heen kunnen veranderen en dus niet perse een vast waarde hebben!

Declaratie en gebruik

```
int variable;
```

```
int variable = 0;
variable = 10;
```

Datatypes

Naam	Keyword	Omschrijving	Minimum	Maximum
Integer	`int`	Gehele getallen	-2147483648	2147483647
Character	`char`	Letters	0	255
Boolean	`bool`	Waar of niet waar	0 (`False`)	1 (`True`)
Floating point	`float`	Kommagetallen		
Double floating point	`double`	Kommagetallen met dubbele precisie		

CASTEN

Operators

Operators zijn symbolen die aangeven dat er een bepaalde wiskundige of logische manipulatie gedaan moet worden.

WISKUNDIG

Operator	Omschrijving
+	getallen bij elkaar optellen
-	getallen van elkaar aftrekken
*	getallen met elkaar vermenigvuldigen
/	Getallen door elkaar delen
%	Rest van een deling

TOEWIJZINGEN

Operator	Omschrijving
+ =	Wijst het resultaat van een toevoeging toe
- =	Wijst het resultaat va een aftrekking toe
* =	Wijst het resultaat van een vermenigvuldiging toe
%	Wijst de rest van een divisie toe
&	Wijst het resultaat van een logische AND toe*
|	Wijst het resultaat van een logische OR toe
!	Geeft aan dat iets, niet, is

AND operator

Een AND operator is een boolean operator. Een boolean heeft maar twee mogelijke resultaten: Waar(1) op niet waar(0). Een boolean heeft verschillende inputs. Bij een simpele boolean zijn er twee inputs.

Een AND Operator is waar (1) wanneer alle inputs ook waar zijn. Even een voorbeeldje: Boolean: Ruben en Hilke hebben allebei blond haar Input 1: Ruben heeft bruin haar Input 2: Hilke heeft blond haar Wanneer we naar de input kijken kunnen we zien dat maar 1 van de twee inputs klopt. Ruben heeft helemaal geen blond haar. Dit houdt in dat de AND operator een onwaar of een 0 terug geeft.

We zouden de input ook anders op kunnen schrjiven: Input 1: Ruben !blond (Zie je wat hier gebeurd, er wordt nu aangegeven dat Ruben geen blond haar heeft. Wat hij wel heeft is onduidelijk maar dat doet er nu niet toe!) Input 2: Hilke heeft blond haar

OR operator

Een OR operator lijkt hele erg op de AND operator. Er is alleen een groot verschil: Om ervoor te zorgen dat de boolean waar (1) terug geeft hoeft er maar 1 van de inputs waar te zijn.

Als we dan weer gaan kijken naar het eerdere voorbeeld: Boolean: Ruben en Hilke hebben allebei blond haar Input 1: Ruben heeft bruin haar Input 2: Hilke heeft blond haar Daar waar het resultaat van de AND operator niet waar (0) was is deze bij de OR operator wel waar (1). 1 van de inputs klopt, Hilke heeft namelijk blond haar.

VERGELIJKINGEN

Operator	Omschrijving
<	Waarde x is kleiner dan waarde y
<=	Waarde x is kleiner of gelijk aan waarde y
>	Waarde x is groter dan waarde x
>=	Waarde x is groter of gelijk aan waarde y
==	Waarde x is waarde y

Scope

Wanneer je een script schrijft heb je zolas eerder vermeld variablen nodig (ja, ik tune toch nog een keertje in op het jeugdtrauma). Nu heb je keuze tussen twee smaakjes van dat trauma eehhhh keuze tussen twee soorten variablen: global variables en local variables

GLOBAL VARIABLES

Global variables zijn variables die gedeclared zijn buiten functies en die global variables kunnen dus gebruikt worden door iedere funcite. Ze zijn zeg maar niet in een hokje gestopt en iedere functie kan aanspraak maken op een global variable. Een global variable is een soort van Albert Heijn, iedereen heeft toegang tot een Albert Heijn en wij mogen naar harte lust appels, Grolsch en andere vernaperingen halen.

LOCAL VARIABLES

Local variables zijn net wat anders. Een local variable wordt gedeclared in een specifieke functie en deze variable kan dan ook alleen maar worden gebruikt binnen die specifieke functie. Het is dus geen Albert Heijn maar de koelkast op Shine. Niet iedereen heeft toegang tot die koelkast maar iedereen op huize Shine mag wel alles in de koelkast gebruiken.

```
int value = 10; // this is a global variable

void setup() {
  int scopedValue; // this variable exists only in this scope
  scopedValue = 11;
}

void loop() {
  // value is accessible here, scopedValue not
}
```

Arrays

Een array is een lijst van waarden. Alle waarden in de array moeten hetzelfde datatype hebben en binnen Arduino is het verplicht om vooraf aan te geven hoe lang de array is, oftewel hoeveel waarden er in passen. Vervolgens kun je de verschillende waarden in de lijst opvragen en aanpassen.

EEN ARRAY AANMAKEN

Je maakt een array aan door aan te geven welk datatype hij heeft en hoe lang de array moet zijn. Bijvoorbeeld een lijst van 6 integers.

```
int integerArray[6];
```

Het is ook mogelijk om meteen waarden in de lijst te stoppen, bijvoorbeeld een lijst met alle weekdagen. Je hoeft dan niet persé de lengte nog op te geven, dat snap de compiler dan wel.

```
String weekDays[]= {"Zondag", "Maandag", "Dinsdag", "Woensdag", "Donderdag", "Vrijdag", "Zaterdag"};
```

WAARDEN UIT EEN ARRAY HALEN

Je kunt een waarde uit een bepaalde plek in de array halen aan de hand van de index van de waarde. Als we bijvoorbeeld de 5e waarde uit de array willen halen doen we dat op de volgende manier.

```
int waarde = integerArray[4];
```

Daarbij moet je er op letten dat het eerste element in de array op index positie 0 staat. Programmeurs beginnen met tellen bij 0, dus het eerste element heeft index 0 en in een array met 6 waarden is de index van het laatste element dus 5.

Een ander aandachtspunt is dat je niet een waarde uit de array kunt opvragen die niet bestaat, bijvoorbeeld bij een array met lengte 6 geeft een poging om de waarde op index 6 op te halen een error, immers deze bestaat niet.

WAARDEN IN EEN ARRAY STOPPEN

Je voegt een waarde aan een array toe op de volgende manier.

```
integerArray[4] = 100;
```

LENGTE VAN DE ARRAY BEPALEN

Mocht je willen weten hoe lang een array is dan kun je de `sizeof(array)` functie gebruiken.

Strings

Strings kun je gebruiken om woorden of zinnen in op te slaan. Eigenlijk zijn Stings een beetje bijzonder, in essentie is een woord of zin namelijk een lijst van letters. Een String bestaat dus eigenlijk uit een array van `char`s in volgorde. En dat maakt dat er wat speciale dingen gelden voor het werken met Strings. Meer informatie daarover vind je in de reference.

EEN STRING AANMAKEN

Je maakt een string aan op de volgende manier.

```
String zin = "Hello World!";
```

Handig om te weten is dat als je aan de string een enter toe wil voegen je daarvoor `\n` kunt gebruiken en een tab is `\t`.

4.2.4 Besluiten maken

We zijn nu aangekomen bij een van de belangrijkste pricipes van programmeren, if-statements. Hiermee kun je bepalen of bepaalde stukken wel of niet uitgevoerd moeten worden. Een if statement heeft een conditie, en als die conditie waar is dan wordt de code uitgevoerd. De conditie kan van alles zijn zoals een boolean variabele, een vergelijking of een combinatie daarvan met and of or statements er tussen.

```
if (condition) {
    // execute this code
}
```

Je kunt ook een alternatief geven voor als de conditie niet waar is door middel van een `else`. Met `else if` kun je code toevoegen die uitgevoerd wordt als de if en alle voorgaande else if condities niet waar zijn en de else if conditie wel.

```
if (condition1) {
    // execute this code if condition 1
} else if (condition2) {
    // execute this code if not condition 1 but condition2
} else {
    // execute this code if neither condition 1 nor condition2
}
```

4.2.5 Loops

Met een loop kun je code herhaald uitvoeren. Er zijn twee verschillende soorten loops, for loops en while loops.

For loop

De code in een for loop wordt zo vaak uitgevoerd als dat je vooraf insteld. In het voorbeeld hieronder is dat 10 keer. In de for loop kun je ook opvragen hoe vaak de loop al uitgevoerd is. In het voorbeeld hieronder kun je dat doen door de waarde van `i` op te vragen.

```
for (int i = 0; i < 10; i++) {
    // execute this code 10 times
}
```

While loop

De while loop wordt net zo lang uitgevoerd totdat de conditie niet meer waar is. Het risico hierbij is dat als de waarde nooit false wordt de code dus tot in de eeuwigheid uitgevoerd wordt. Daar moet je dus rekening mee houden.

```
while (condition) {
    // execute this code until condition is false
}
```

4.2.6 Seriële communicatie

Seriële communicatie kan gebruikt worden om berichten van de microcontroller naar de computer te sturen. Deze kun je dan in de seriële monitor uitlezen. Allereerst moet je de bitrate instellen, in het voorbeeld wordt deze ingesteld op 9600. Belangrijk is dat je seriële monitor en je microcontroller dezelfde bitrate gebruiken, anders zijn de berichten onleesbaar. Met de

`Serial.begin(bautrate);` functie kun je in de `setup()` een seriële verbinding opzetten. Vervolgens kun je met de `Serial.println(message);` functie een bericht naar de monitor sturen.

```
#include <Arduino.h>

void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

void loop() {
  // Print the line "Hello World!"
  Serial.println("Hello World!");

  // wait for 1 second before printing again
  delay(1000);
}
```

4.2.7 Functies

Functies kun je gebruiken om je code meer structuur te geven of om stukken code her te gebruiken. Je moet een functie eerst definiëren voordat je hem kunt gebruiken.

```
void function() {
    // execute code here
}
```

Een `void` functie geeft geen waarde terug maar je kunt een funtie ook een datatype geven en een waarde terug geven door middel van `return`. Verder is het mogelijk om parameters aan de functie toe te voegen aan de functie zodat je waarden aan de funtie mee kunt geven. In het voorbeeld hieronder wordt een optel functie getoond die de twee parameters bij elkaar opteld en terug geeft. Na het uitvoeren van deze code is de waarde van de variabele `result` dus 3.

```
int add(int param1, int param2) {
    return param1 + param2;
}

int result = add(1, 2);
```

4.2.8 Conclusie

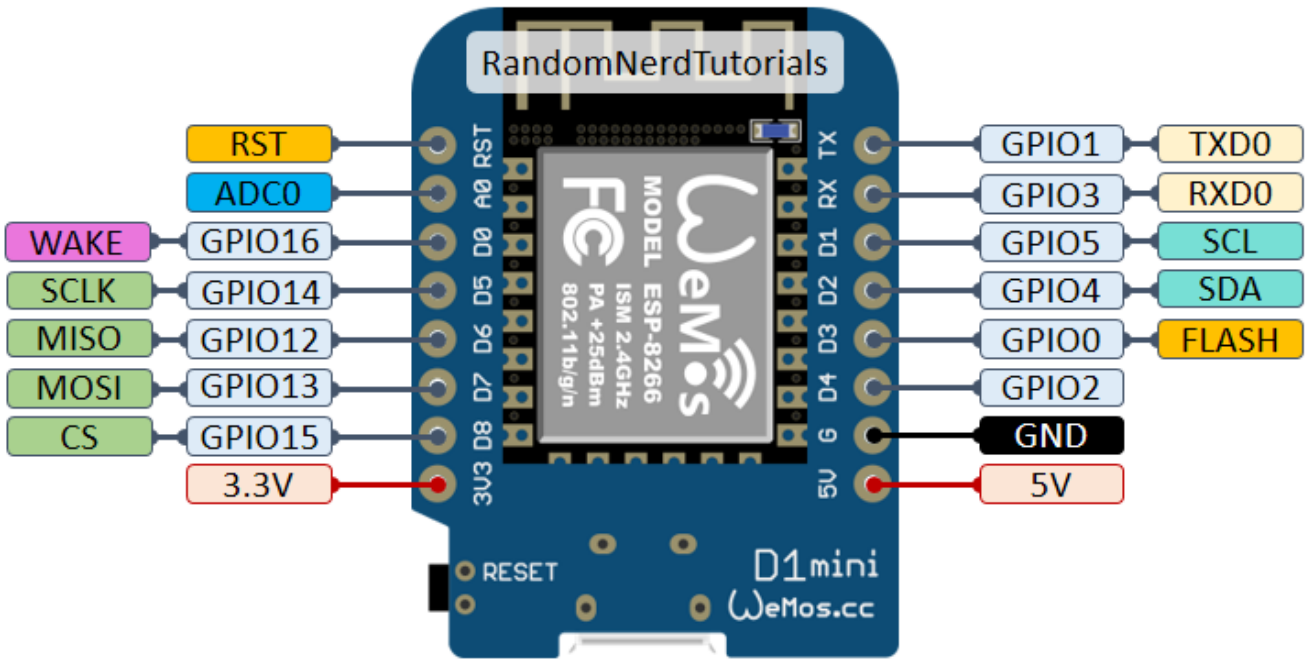
Als je tot in detail informatie wilt over hoe programmeren met Arduino werkt kun je altijd een blik werpen in de reference. Maar er zijn ook veel handleidingen online te vinden. We hopen dat je dit een overzicht geeft waarmee je aan de slag kunt. Kijk ook eens op de meer informatie pagina over handige websites over hoe je kunt programmeren met Arduino en de ESP.

4.3 Input & Output

Het mooie aan microcontrollers is natuurlijk dat je er elektronische componenten mee kunt aansturen en uitlezen. In dit hoofdstuk zullen we beschrijven hoe je de input en output van de microcontroller kunt gebruiken.

4.3.1 Pinout

De ESP8266 die wij voor dit project gebruiken is gemonteerd op een WEMOS D1 mini development board. Daarop zijn een aantal pinnen beschikbaar zoals te zien in de onderstaande afbeelding. We zullen de functionaliteiten van elke pin hieronder uitleggen.



- `D0 tm D8` De 9 digitale pinnen kunnen gebruikt worden als digitale input of output. Op `D0` na ondersteunen ze allemaal PWM.
- `A0` Is een analoge input pin.
- `RX & TX` Kunnen gebruikt worden voor Seriële communicatie.
- `GND` Is een ground pin.
- `3V3` Geeft een spanning van 3,3V waarop de ESP draait.
- `5V` Geeft een spanning van 5V direct vanaf de USB voeding.

Als je meer informatie wilt over de werking en mogelijkheden van elke pin kun je kijken in de pin reference.

4.3.2 Digitale Input

Om een digitale pin van de ESP als input te gebruiken zodat je er bijvoorbeeld een knopje op aan kunt sluiten zul je in de `setup()` van je programma moeten aangeven welke pin als output behandeld moet worden. Dit doe je met de `pinMode(pinNumber, mode)` functie waarbij je het pinnummer en de modus opgeeft.

```
1  pinMode(D1, INPUT);
```

Vervolgens kun je de waarde van de input uitlezen met `digitalRead(pinNumber)`. De ESP draait op 3.3 volt dus wanneer er een spanning van ongeveer 3.3 volt op de pin staat wordt er een `True` teruggegeven. Is het voltage rond de 0 volt (oftewel ground) dan wordt er een `False` terug gegeven.

```
boolean value = digitalRead(D1);
```

4.3.3 Digitale Output

Het is ook mogelijk een digitale pin als output te gebruiken. Daarvoor zet je in de `setup()` de pinMode van die pin op `OUTPUT`.

```
pinMode(D1, OUTPUT);
```

Vervolgens kun je met `digitalWrite(pinNumber, state)` de output van de pin aanpassen. Met `HIGH` komt er 3.3 volt op de pin te staan en met `LOW` wordt de pin verbonden met ground.

```
digitalWrite(D1, HIGH);
```

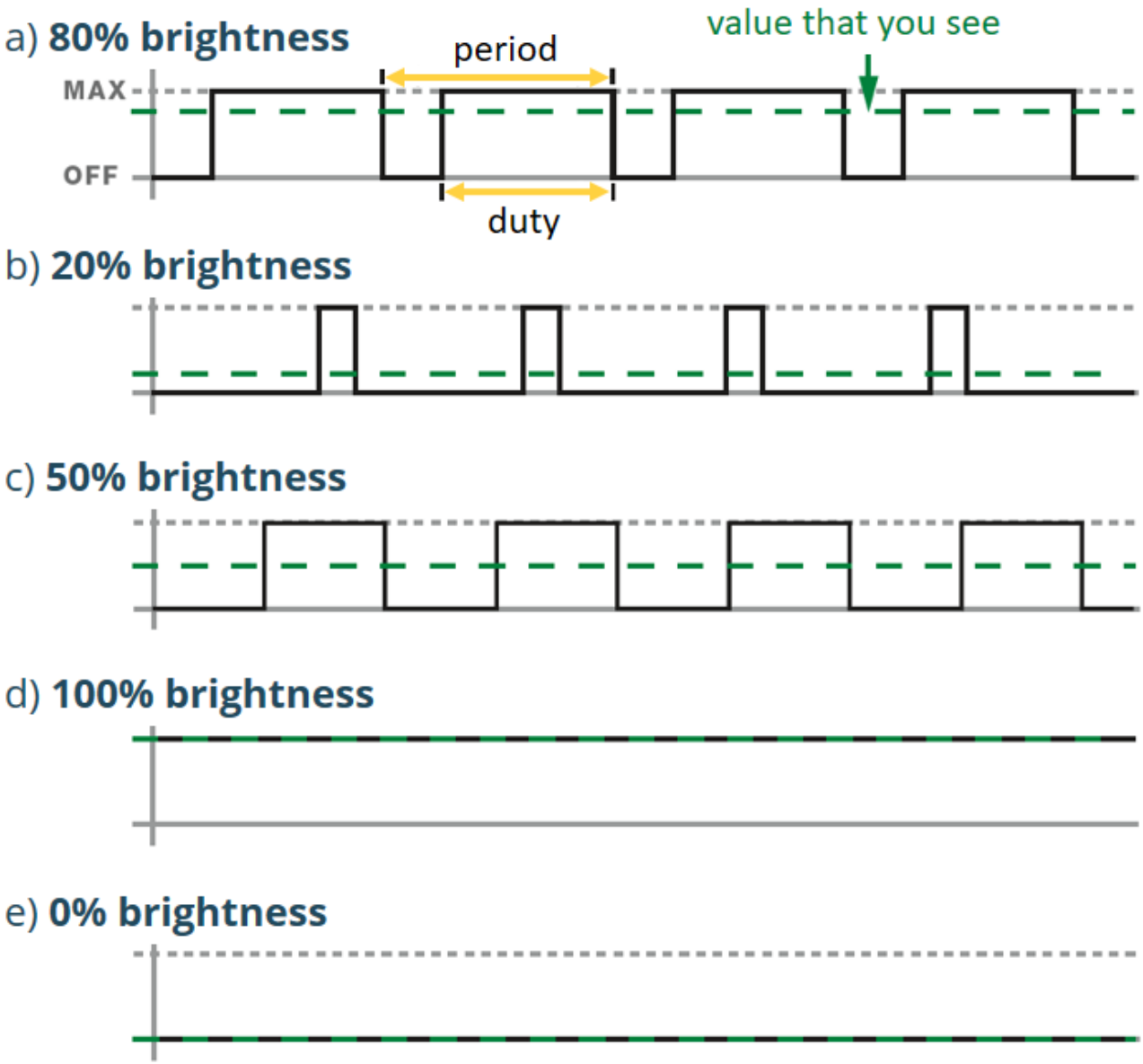
4.3.4 Analoge Input

De ESP heeft één analoge pin, `A0`, waarvan je de waarde uit kunt lezen met `analogRead(pinNumber)`. Dit geeft een getal tussen de 0 en de 1023 terug waarbij 0 overeen komt met een verbinding met de ground en 1023 met 3.3 volt.

```
int value = analogRead(A0);
```

4.3.5 PWM Output

Het is mogelijk om met een digitale pin een analoge output te simuleren. Dat doen we door de pin heel snel aan en uit te zetten, het zogenomede Pulse Width Modulation (PWM). Door de tijd dat de pin aan staat in een periode te varieren kunnen we het schijnbare voltage variëren. Dit noemen we de dutycycle aanpassen. PWM wordt vooral gebruikt om de helderheid van ledlampjes aan te passen. Onze ogen waanemen namelijk het gemiddelde van de helderheid. In de onderstaande afbeelding zie je een voorbeeld van hoe dit werkt.



Als je PWM op een digitale pin wilt gebruiken moet je hem eerst als output instellen in de `setup()`.

```
1   pinMode(D1, OUTPUT);
```

Vervolgens roep je de `analogWrite(pinNumber, value)` functie aan. De `value` mag een getal tussen de 0 en de 255 zijn. Het volgende voorbeeld geeft dus een PWM signaal op 50% via pin `D1`.

```
1   analogWrite(D1, 127);
```

4.4 Ledring

Om de kleuren en helderheid van de leds op de ledring aan te passen kun je gebruik maken van de Adafruit Neopixel library. Om deze te installeren ga je naar **Sketch > Library > Manage Libraries** en zoek je naar Adafruit Neopixel. Deze library geeft handige functies waarmee de leds aangestuurd kunnen worden op led ringen maar ook ledstrips.

4.4.1 Functies

De library heeft de volgende functies.

- `begin()` initialiseert het ledring object waarna het gebruikt kan worden.
- `show()` schrijft de wijzigingen aan de pixel kleuren naar de ledring voor weergave.
- `setPixelColor(n, red, green, blue)` zet de kleur van de pixel.
- `fill(color, first, count)` zet de kleur van meerdere achtereenvolgende pixels.
- `Color(red, green, blue)` converteerd losse waarden voor rood, groen en blauw naar een enkele kleurwaarde.
- `ColorHSV(hue, saturation, value)` converteerd losse waarden voor hue, saturation en value naar een enkele kleurwaarde.
- `getPixelColor(n)` geeft de kleur van een pixel.
- `setBrightness(brightness)` zet de helderheid van de ledring met een waarde tussen de 0 en de 255.
- `getBrightness()` geeft de helderheid van de ledring.
- `clear()` verwijdert alle ingestelde kleuren van elke pixel.
- `numPixels()` geeft het aantal pixels in de ledring.
- `gamma32(color)` geeft een gamma gecorigeerde kleurwaarde terug.

4.4.2 Voorbeeld

```
#include <Adafruit_NeoPixel.h>

// Which pin on the ESP is connected to the NeoPixels?
#define LED_PIN D4

// How many leds are there on the ring?
#define LED_COUNT 60

// Declare our NeoPixel strip object:
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);
// Argument 1 = Number of pixels in NeoPixel strip
// Argument 2 = Pin number (most are valid)
// Argument 3 = Pixel type flags, add together as needed:
//   NEO_KHZ800  800 KHz bitstream (most NeoPixel products w/WS2812 LEDs)
//   NEO_KHZ400  400 KHz (classic 'v1' (not v2) FLORA pixels, WS2811 drivers)
//   NEO_GRB     Pixels are wired for GRB bitstream (most NeoPixel products)
//   NEO_RGB     Pixels are wired for RGB bitstream (v1 FLORA pixels, not v2)
//   NEO_RGBW    Pixels are wired for RGBW bitstream (NeoPixel RGBW products)

void setup() {
  strip.begin();           // INITIALIZE NeoPixel strip object (REQUIRED)
  strip.show();            // Turn OFF all pixels ASAP
  strip.setBrightness(50); // Set BRIGHTNESS to about 1/5 (max = 255)
}

void loop() {
  // Fill along the length of the strip in various colors...
  colorWipe(strip.Color(255,   0,   0), 50); // Red
  colorWipe(strip.Color(  0, 255,   0), 50); // Green
  colorWipe(strip.Color(  0,   0, 255), 50); // Blue

  // Do a theater marquee effect in various colors...
  theaterChase(strip.Color(127, 127, 127), 50); // White, half brightness
  theaterChase(strip.Color(127,   0,   0), 50); // Red, half brightness
  theaterChase(strip.Color(  0,   0, 127), 50); // Blue, half brightness

  rainbow(10);             // Flowing rainbow cycle along the whole strip
  theaterChaseRainbow(50); // Rainbow-enhanced theaterChase variant
}

// Fill strip pixels one after another with a color. Strip is NOT cleared
// first; anything there will be covered pixel by pixel. Pass in color
// (as a single 'packed' 32-bit value, which you can get by calling
// strip.Color(red, green, blue) as shown in the loop() function above),
// and a delay time (in milliseconds) between pixels.
void colorWipe(uint32_t color, int wait) {
  for(int i=0; i<strip.numPixels(); i++) { // For each pixel in strip...
    strip.setPixelColor(i, color);         //  Set pixel's color (in RAM)
    strip.show();                          //  Update strip to match
    delay(wait);                           //  Pause for a moment
  }
}

// Theater-marquee-style chasing lights. Pass in a color (32-bit value,
// a la strip.Color(r,g,b) as mentioned above), and a delay time (in ms)
// between frames.
void theaterChase(uint32_t color, int wait) {
  for(int a=0; a<10; a++) {  // Repeat 10 times...
    for(int b=0; b<3; b++) { //  'b' counts from 0 to 2...
      strip.clear();         //   Set all pixels in RAM to 0 (off)
      // 'c' counts up from 'b' to end of strip in steps of 3...
      for(int c=b; c<strip.numPixels(); c += 3) {
        strip.setPixelColor(c, color); // Set pixel 'c' to value 'color'
      }
      strip.show(); // Update strip with new contents
      delay(wait);  // Pause for a moment
    }
  }
}

// Rainbow cycle along whole strip. Pass delay time (in ms) between frames.
void rainbow(int wait) {
  // Hue of first pixel runs 5 complete loops through the color wheel.
  // Color wheel has a range of 65536 but it's OK if we roll over, so
  // just count from 0 to 5*65536. Adding 256 to firstPixelHue each time
  // means we'll make 5*65536/256 = 1280 passes through this outer loop:
  for(long firstPixelHue = 0; firstPixelHue < 5*65536; firstPixelHue += 256) {
    for(int i=0; i<strip.numPixels(); i++) { // For each pixel in strip...
      // Offset pixel hue by an amount to make one full revolution of the
      // color wheel (range of 65536) along the length of the strip
      // (strip.numPixels() steps):
      int pixelHue = firstPixelHue + (i * 65536L / strip.numPixels());
      // strip.ColorHSV() can take 1 or 3 arguments: a hue (0 to 65535) or
      // optionally add saturation and value (brightness) (each 0 to 255).
      // Here we're using just the single-argument hue variant. The result
      // is passed through strip.gamma32() to provide 'truer' colors
      // before assigning to each pixel:
      strip.setPixelColor(i, strip.gamma32(strip.ColorHSV(pixelHue)));
    }
    strip.show(); // Update strip with new contents
    delay(wait);  // Pause for a moment
  }
}

// Rainbow-enhanced theater marquee. Pass delay time (in ms) between frames.
void theaterChaseRainbow(int wait) {
  int firstPixelHue = 0;     // First pixel starts at red (hue 0)
  for(int a=0; a<30; a++) {  // Repeat 30 times...
    for(int b=0; b<3; b++) { //  'b' counts from 0 to 2...
      strip.clear();         //   Set all pixels in RAM to 0 (off)
      // 'c' counts up from 'b' to end of strip in increments of 3...
      for(int c=b; c<strip.numPixels(); c += 3) {
        // hue of pixel 'c' is offset by an amount to make one full
        // revolution of the color wheel (range 65536) along the length
```

4.4.3 Werking

Allereerst moet de library toegevoegd worden aan de applicatie.

```
#include <Adafruit_NeoPixel.h>
```

Vervolgens slaan we op aan welke pin de ledring verbonden is en hoeveel pixels er op de ring zitten, in ons geval pin `D4` en 60 pixels. Daarna maken we een Neopixel strip object aan.

```
#define LED_PIN D4
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);
```

setup()

In de `setup()` initializeren we het strip object door de `begin()` functie aan te roepen en zetten we alle pixels uit door de `show()` functie aan te roepen.

```
void setup() {
  strip.begin();
  strip.show();
}
```

Pixels een kleur geven

De kleur van een pixel kan ingesteld worden door middel van de `setPixelColor(n, red, green, blue)`. Daar bij is n het nummer van de pixel in de ring en zijn de kleuren gegeven als een getal tussen de 0 en 255. Het volgende voorbeeld zet de kleur van de 12e pixel naar magenta.

```
strip.setPixelColor(11, 255, 0, 255);
```

Een alternatief is om de kleur van de pixel in te stellen met een kleurwaarde met de `setPixelColor(n, color)` functie. Daarvoor kun je de waarde van een kleur in een 32 bit getal opslaan zodat je hem later kunt hergebruiken. Hiervoor kun je de `Color(red, green, blue)` functie gebruiken. Het volgende voorbeeld slaat de kleur magenta op in een variabele en zet vervolgens de kleur van de 12e pixel naar magenta.

```
uint32_t magenta = strip.Color(255, 0, 255);
strip.setPixelColor(11, magenta);
```

> **Let op! `setPixelColor()` heeft niet meteen effect op de leds**
>
> De ingestelde kleuren moeten eerst naar de ledring verstuurd worden. Daarvoor kun je de `show()` functie gebruiken.

De `show()` functie zorgt ervoor dat de voor de pixels ingestelde kleuren weergeven op de ledring.

```
strip.show();
```

Meerdere pixels een kleur geven

Je kunt meerdere pixels dezelfde kleur geven met de `fill(color, first, count)` functie. Daarbij is `color` een 32-bit kleur waarde, is `first` de eerste pixel die de kleur moet krijgen en is `count` het aantal pixels dat de kleur moeten krijgen. In het volgende voorbeeld worden de pixels 4 tot en met 8 de kleur magenta gegeven.

```
uint32_t magenta = strip.Color(255, 0, 255);
strip.fill(magenta, 3, 5);
```

De `clear()` functie kan gebruikt worden om alle pixels uit te zetten.

```
strip.clear();
```

Met `numPixels()` kun je het aantal pixels in de ring opvragen. Dit kun je gebruiken in een for loop om alle pixels een kleur te geven. Het volgende voorbeeld maakt een kleurverloop van rood naar blauw over de hele ring.

```
for (int i = 0; i < strip.numPixels(); i++) {
  int offset = (int) (255 / strip.numPixels()) * i;
  strip.setPixelColor(i, 255 - offset, 0, offset);
}
```

Helderheid

De helderheid van de hele ledring kan met `setBrightness(value)` ingesteld worden. Daarbij is `value` een waarde tussen de 0 voor uit en 255 voor maximale helderheid. Om de ledring op een kwart helderheid te zetten kun je dus de volgende code gebruiken:

```
strip.setBrightness(64);
```

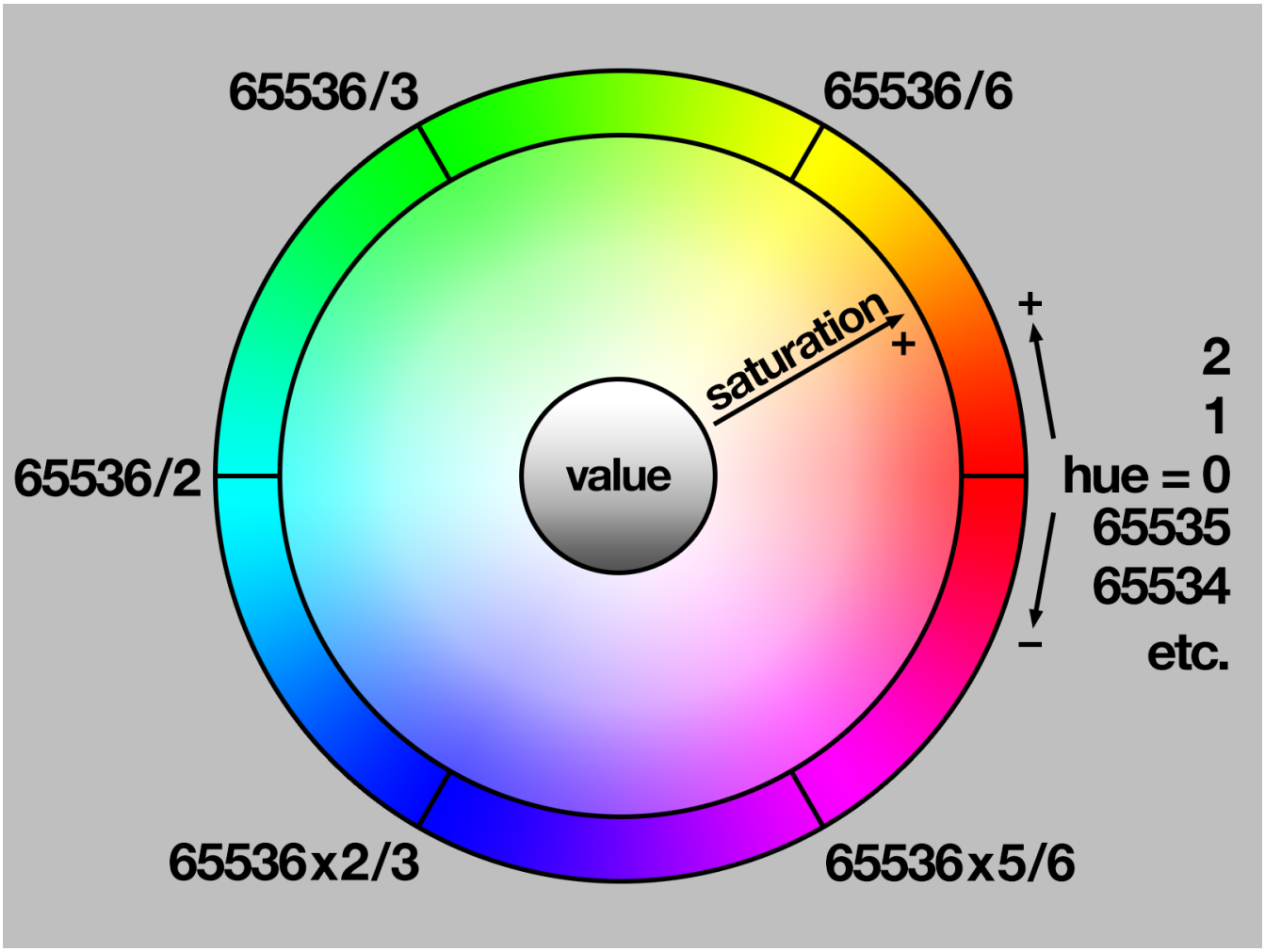
> **Let op! `setBrightness()` heeft niet meteen effect op de leds**
>
> De ingestelde helderheid moeten eerst naar de ledring verstuurd worden. Daarvoor kun je de `show()` functie gebruiken.

> **`setBrightness()` is eigenlijk bedoeld om maar één keer aangeroepen te worden in de `setup()`**
>
> Het is dus eigenlijk niet de bedoeling om hem voor animaties te gebruiken. Het is beter om met eigen logica de helderheid van je pixels aan te passen als je hier animaties mee wilt maken. Dit komt omdat de functie de waarden van je pixel data in het RAM aanpast en daarmee informatie verloren gaat, het is een "lossy" operatie.

HSV kleuren

De Neopixel library ondersteunt ook het gebruik van kleuren in de "HSV" (hue-saturation-value) kleurruimte als alternatief voor de gebruikelijke RGB (red-green-blue). Voor sommige effecten is dit veel gebruiksvriendelijker, bijvoorbeeld voor het regenboog effect (zeer populair onder creaters). Een HSV kleur kan met de `ColorHSV()` opgeslagen worden als RGB waarde.

```
uint32_t rgbcolor = strip.ColorHSV(hue, saturation, value);
```



Daarbij zijn de volgende parameters van belang:

- `hue` is een 16-bit nummer dat start bij 0 voor rood en de kleurcirkel via geel (65536/6), groen (65536/3), cyaan (65536/2), blauw (655362/3) en magenta (655365/6) weer bij rood aan komt (65536).
- `saturation` is een 8-bit nummer dat de verzadiging van de kleur aangeeft waarbij 0 onverzadigd is en 255 maximale verzadiging.
- `value` is een 8-bit nummer dat de helderheid van de kleur aangeeft waarbij 0 zwart is en 255 maximale helderheid.

Als je alleen een pure kleur wilt met maximale verzadiging en helderheid dan kun je de functie ook aanroepen met alleen de hue:

```
uint32_t rgbcolor = strip.ColorHSV(hue);
```

GAMMA CORRECTIE

Als je veel met kleuren werkt zul je merken dat wanneer je faded tussen kleuren het soms lijkt alsof ze overmatig helder of flets er uit zien. Dit komt omdat nummeriek de kleurwaarden goed zijn maar de perceptie van onze ogen iets anders is. Daarvoor kun je zogenaamde gamma correctie gebruiken, hoe dat precies werkt wordt hier uitgelegd. De `gamma32(color)` functie neemt een RGB waarde en corrigeert deze zodat hij optisch correct is.

```
uint32_t rgbcolor = strip.gamma32(strip.ColorHSV(hue, sat, val));
```

EEN REGENBOOG MAKEN

Stel je wilt je ledstrip voorzien van een regenboogeffect (zeer belangrijk, dit wil je weten) dan doe je dat als volgt:

```
for(int i = 0; i < strip.numPixels(); i++) {
  int hue = i * 65536L / strip.numPixels();
  uint32_t color = strip.gamma32(strip.ColorHSV(hue));
  strip.setPixelColor(c, color); // Set pixel 'c' to value 'color'
}
```

4.4.4 Meer informatie

Voor meer informatie kun je naar de github repo van de library of vind je in de Neopixel Uberguide.

4.5 Datum en tijd

Om de huidige datum en tijd te gebruiken binnen je applicatie kun je gebruik maken van de NTPClient library. Om deze te installeren ga je naar **Sketch > Library > Manage Libraries** en zoek je naar NTPClient. Deze library verbint via Wifi met een server die de huidige datum en tijd kan vertellen. Een voorbeeld van een zogenaamde NTP server is pool.ntp.org.

4.5.1 Functies

De library heeft de volgende functies.

- `getDay()` geeft een nummer dat overeenkomt met de dagen van de week waarbij 0 zondag is.
- `getHours()` geeft een nummber met het huidige uur in 24 uurs formaat.
- `getMinutes()` geeft een nummer met de huidige minuut.
- `getSeconds()` geeft een nummer met de huidige seconde.
- `getEpochTime()` geeft een unsigned long met de huidige epoch tijd in seconden. Dat is het aantal seconden sinds middernacht 1 januari 1970 in GMT.
- `getFormattedTime()` geeft een String met de tijd in HH:MM:SS format.

Er is geen functie om de datum te berekenen maar dit is wel af te leiden uit de epoch.

4.5.2 Voorbeeld

In het onderstaande voorbeeld worden alle functionaliteiten van de NTPclient library gedemonstreerd.

```
/*
  Rui Santos
  Complete project details at https://RandomNerdTutorials.com/esp8266-nodemcu-date-time-ntp-client-server-arduino/

  Permission is hereby granted, free of charge, to any person obtaining a copy
  of this software and associated documentation files.

  The above copyright notice and this permission notice shall be included in all
  copies or substantial portions of the Software.
*/

#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>

// Replace with your network credentials
const char *ssid     = "REPLACE_WITH_YOUR_SSID";
const char *password = "REPLACE_WITH_YOUR_PASSWORD";

// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");

//Week Days
String weekDays[7]={"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"};

//Month names
String months[12]={"January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"};

void setup() {
  // Initialize Serial Monitor
  Serial.begin(115200);

  // Connect to Wi-Fi
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

// Initialize a NTPClient to get time
  timeClient.begin();
  // Set offset time in seconds to adjust for your timezone, for example:
  // GMT +1 = 3600
  // GMT +8 = 28800
  // GMT -1 = -3600
  // GMT 0 = 0
  timeClient.setTimeOffset(0);
}

void loop() {
  timeClient.update();

  unsigned long epochTime = timeClient.getEpochTime();
  Serial.print("Epoch Time: ");
  Serial.println(epochTime);

  String formattedTime = timeClient.getFormattedTime();
  Serial.print("Formatted Time: ");
  Serial.println(formattedTime);

  int currentHour = timeClient.getHours();
  Serial.print("Hour: ");
  Serial.println(currentHour);

  int currentMinute = timeClient.getMinutes();
  Serial.print("Minutes: ");
  Serial.println(currentMinute);

  int currentSecond = timeClient.getSeconds();
  Serial.print("Seconds: ");
  Serial.println(currentSecond);

  String weekDay = weekDays[timeClient.getDay()];
  Serial.print("Week Day: ");
  Serial.println(weekDay);

  //Get a time structure
  struct tm *ptm = gmtime ((time_t *)&epochTime);

  int monthDay = ptm->tm_mday;
  Serial.print("Month day: ");
  Serial.println(monthDay);

  int currentMonth = ptm->tm_mon+1;
  Serial.print("Month: ");
  Serial.println(currentMonth);

  String currentMonthName = months[currentMonth-1];
  Serial.print("Month name: ");
  Serial.println(currentMonthName);

  int currentYear = ptm->tm_year+1900;
  Serial.print("Year: ");
  Serial.println(currentYear);

  //Print complete date:
  String currentDate = String(currentYear) + "-" + String(currentMonth) + "-" + String(monthDay);
  Serial.print("Current date: ");
  Serial.println(currentDate);

  Serial.println("");
```

4.5.3 Werking

Allereerst moeten de benodigde libraries toegevoegd worden.

```
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
```

Vervolgens maken we twee variabelen aan waarin we de naam van het wifi netwerk en het wachtwoord in opslaan.

```
const char *ssid     = "NETWERKNAAM";
const char *password = "WACHTWOORD";
```

Daarna maken we een NTP client aan om de datum en tijd mee op te halen.

```
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");
```

setup()

In de `setup()` maken we verbinding met het internet. Voor het gemak printen we de status hiervan naar de serialmonitor, dit zo je kunnen weglaten.

```
Serial.begin(9600);

Serial.print("Connecting to ");
Serial.println(ssid);

WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}
```

Hierna initializeren we de NTP client.

```
timeClient.begin();
```

TIJDSZONE

Je kunt de huidige tijdszone instellen met de `setTimeOffset()` methode. Hier geef je het aantal seconden verschil met GMT, Nederlandse wintertijd is 3600 en zomertijd is 7200.

```
timeClient.setTimeOffset(3600);
```

loop()

In de `loop()` roepen we de `update()` functie aan om de huidige datum en tijd op te halen van de NTP server.

```
timeClient.update();
```

TIJD OPHALEN

Als we de huidige epoch tijd willen printen dan kan dat als volgt.

```
unsigned long epochTime = timeClient.getEpochTime();
Serial.print("Epoch Time: ");
Serial.println(epochTime);
```

De `getFormattedTime()` functie kan de tijd in een geformatte string weergeven.

```
String formattedTime = timeClient.getFormattedTime();
Serial.print("Formatted Time: ");
Serial.println(formattedTime);
```

Om de uren, minuten of seconden te bepalen kun je gebruik maken van de `getHours()`, `getMinutes()` en de `getSeconds()` functies.

```
int currentHour = timeClient.getHours();
Serial.print("Hour: ");
Serial.println(currentHour);

int currentMinute = timeClient.getMinutes();
Serial.print("Minutes: ");
Serial.println(currentMinute);

int currentSecond = timeClient.getSeconds();
Serial.print("Seconds: ");
Serial.println(currentSecond);
```

DATUM OPHALEN

De `getDay()` functie geeft alleen een nummer tussen de 0 en 6, waarbij zondag 0 is, om de dag aan te geven. Als je graag de naam van de dag wilt printen dan kun je deze uit een lijst halen als volgt.

```
String weekDays[7]={"Zondag", "Maandag", "Dinsdag", "Woensdag", "Donderdag", "Vrijdag", "Zaterdag"};

String weekDay = weekDays[timeClient.getDay()];
Serial.print("Week Day: ");
Serial.println(weekDay);
```

Er is geen standaard functie om de huidige datum op te halen. Het is wel mogelijk dit te bepalen door middel van een time structure. Meer informatie hierover vind je in de c++ referentie of kijk in het voorbeeld.

4.6 Meer informatie

Stel je wilt meer informatie over programmeren in Arduino of met de ESP8266 dan zijn hier een aantal bronnen met meer informatie.

- Arduino reference Alle documentatie over de arduino library en de functies die beschikbaar zijn.
- Random Nerd Tutorials Een site met duidelijke uitleg en ook veel projecten met onder andere de ESP8266.

5. Projecten

5.1 Projectoverzicht

Nu je de ledring in elkaar gezet hebt en weet hoe je hem moet programmeren is het tijd er wat leuks mee te maken. We hebben een paar handleidingen en wat ideeën op een rij gezet.

5.1.1 Projecthandleidingen

We hebben een aantal projecten die je met de ledring kunt maken. Deze zijn in oplopende mate van complexiteit, beginnend met een nachtlampje en eindigend met een koppeling met je agenda. We zullen stap voor stap door de code heen lopen en aan het eind zullen we de volledige code weergeven van het project. Heb je dus geen zin om zelf te programmeren dan kun je dus altijd onze code gebruiken.

Beginners: Nachtlampje

Voor wie geen ervaring heeft met Arduino en programmeren is dit een mooi startproject. Door middel van het knopje kun je de ledring aan en uit zetten. De potentiometer bepaalt hoe fel de ledring is.

Gemiddeld: Wakeuplight

We breiden het nachtlampje uit zodat het vanaf een vooraf ingestelde tijd langzaam aan gaat tot de maximale helderheid die met de potentiometer is ingesteld. Om de tijd te bepalen verbinden we met een wifi netwerk.

Gevorderd: Klok

De ledring heeft exact 60 ledjes, en in het project met de wakeuplight hebben we geleerd hoe we de tijd van het internet kunnen ophalen. Logischerwijs is het dus ook mogelijk om een klok te maken. Dat is wat we in dit project zullen doen.

Expert: Agenda

Via de internetverbinding is het ook mogelijk om contact te leggen met andere webservers. In dit project zullen we de tijd van de eerstvolgende afspraak in een online agenda ophalen. En daar kunnen we leuke dingen mee doen. We zullen een teller maken die aftelt naar je eerstvolgende afspraak. Maar met een kleine aanpassing kun je een wakeuplight maken dat automatisch de goede tijd kiest op basis van je agenda, of de teller combineren met een klok.

5.1.2 Doe je eigen project

We zijn uiteraard heel benieuwd welke leuke andere ideeën je hebt met deze ledring. Misschien maak je wel een statusbalk waarmee je kunt zien hoelang het nog duurt totdat de frituur klaar is. Of koop je een luchtkwaliteitssensor en weergeef je de luchtkwaliteit in je studeerkamer. Laat het ons vooral weten en dan zullen we je project aan deze pagina toevoegen. De volgende ideeën en voorbeelden zijn al ingestuurd.

Pixel chaser

Probeer op het juiste moment op het knopje te drukken in dit spel waarmee je je reactietijd kunt meten. Is optioneel ook als drankspel te spelen. Een beschrijving hoe je dit doet vind je in deze video: https://www.youtube.com/watch?v=hjqtJA7gVQ0. De aangepaste code voor de ESP8266 vind je in de repository van deze workshop. Om deze code uit te voeren zul je de OneButton library moeten installeren, dit kan op dezelfde manier als beschreven voor de Neopixel library in het Software hoofdstuk.

Aansturen met app

Het is mogelijk om de ledring aan te sturen door middel van een app. Hiervoor kun je de WLED library installeren op de microcontroller. Hoe je dat doet vind je hier https://github.com/Aircoookie/WLED/wiki. Vervolgens kun je de Android of iOS app gebruiken om de kleur van de ledring aan te passen en animaties op de ring te weergeven. Via WLED is het mogelijk om de ledring aan veel andere diensten te koppelen zoals Philips Hue en Amazon Alexa.

Koppelen met Google Home

Met wat extra stappen is het mogelijk om de ledring aan te sturen via Google Home. Door de beperkingen in Google Home is de functionaliteit wat minimaal en is het redelijk wat werk om op te zetten. Hoe je dit precies doet lees je in de handleiding WLED met IFTTT en Google Home.

Weerbericht

Door een koppeling met buienradar wordt weergeven wat de weersverwachting van vandaag is. Voor dit project is helaas nog geen voorbeeldcode beschikbaar.

Spotify discolicht

Hoe vet is het als je een discolamp hebt die automatisch mee knippert op het tempo van je huidige Spotify liedje. Via de spotify api kun je het tempo, de dansbaarheid en het energielevel van het huidige liedje ophalen en daar de ledring op aanpassen. Voor dit project is helaas nog geen voorbeeldcode beschikbaar.

5.2 Nachtlampje

Welkom bij deze handleiding om een nachtlampje te maken. Alle achtergrondinformatie over de functies en methodes die we gaan gebruiken kun je in het software hoofdstuk vinden. We zullen stap voor stap door de code heen lopen en afsluiten met een voorbeeld van de volledige code. Maar nu eerst een overzicht van de werking.

We zullen in het programma beginnen met alle instellingen die nodig zijn om de ledstrip aan te sturen en het knopje en de potentiometer uit te lezen. Vervolgens zullen we in de `loop()` bepalen of het knopje ingedrukt wordt en het lampje aan of uit zetten. Als het lampje aan moet staan dan zullen we de potentiometer uitlezen hoe fel dat moet zijn en de ledring met die felheid aanzetten.

5.2.1 Voorbereiding

We beginnen met het importeren van de Neopixel library die we nodig hebben om de ledring mee aan te sturen.

```
#include <Adafruit_NeoPixel.h>
```

Daarna definiëren we variabelen waarin we opslaan aan welke pinnen we het knopje, de potentiometer en ledring aangesloten hebben.

```
#define POT_PIN A0
#define BTN_PIN D8
#define LED_PIN D4
```

Vervolgens maken we een strip object aan dat de communicatie met de ledring zal verzorgen.

```
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);
```

Tot slot maken we een variabele aan waarin we opslaan of de ledring aan of uit moet staan.

```
boolean lightOn = true;
```

5.2.2 Setup

Vervolgens in de `setup()` starten we de ledstrip op met de `begin()` functie en zetten hem aan met de `show()` functie. Als je nu niet de `show()` functie zou aanroepen zul je zien dat een of meerdere leds op de ring willekeurig aan zullen staan. Aansluitend zorgen we er voor dat de pin waaraan het knopje is verbonden als input uitgelezen moet worden door middel van de `pinMode()` functie.

```
void setup() {
  // We starten de ledstrip op en zetten hem aan
  strip.begin();
  strip.show();

  // We definiëren dat het knopje als input gebruik moet worden
  pinMode(BTN_PIN, INPUT);
}
```

5.2.3 Loop

In de `loop()` staat alle code die herhaald uitgevoerd moet worden. Hierin zullen we bepalen of het knopje ingedrukt wordt en of de ledring aan moet en hoe helder.

Indrukken knopje

Met de `digitalRead()` functie kunnen we bepalen of het knopje ingedrukt wordt of niet. Vervolgens kunnen we in `lightOn` opslaan of de ring aan of uit moet staan. Als hij aan staat moet hij uit en vice-versa. Dus dan komen we tot de volgende code.

```
// Als het knopje ingedrukt wordt moet het lampje aan of uit gaan
if (digitalRead(BTN_PIN)) {
  // We veranderen de waarde naar het tegenovergestelde van zijn vorige waarde, aan wordt dus uit en uit wordt aan
  lightOn = !lightOn;
}
```

DEBOUNCEN

Als je bovenstaande code zou gebruiken zul je merken dat de ledring willekeurig zal knipperen en zodra je het knopje loslaat hij ofwel aan ofwel uit staat. Dat komt doordat er twee problemen zijn. Allereerst bouncing, dat is dat wanneer je het knopje indrukt hij niet meteen volledig contact maakt maar een beetje "stuitert". Vanuit de code gezien heeft dat het effect alsof je heel snel achter elkaar het knopje indrukt. De makkelijkste oplossing is om even te wachten voordat we verder gaan om de bounce periode voorbij te gaan. Daarvoor kun je de `delay()` functie gebruiken. Persoonlijk gebruik ik vaak een delay van 300 miliseconden, dan weet je zeker dat het bouncen wel voorbij is.

Het andere probleem is dat het zo kan zijn dat iemand het knopje ingedrukt houd. In dat geval zal de ring gaan knipperen, en dat is ook een beetje raar. Dus na het debouncen is het slim om te wachten totdat de gebruiker het knopje losgelaten heeft.

Om deze twee punten op te lossen komen we tot de volgende code.

```
// Als het knopje ingedrukt wordt moet het lampje aan of uit gaan
if (digitalRead(BTN_PIN)) {
  // We veranderen de waarde naar het tegenovergestelde van zijn vorige waarde, aan wordt dus uit en uit wordt aan
  lightOn = !lightOn;

  // We wachten heel even om rekening te houden met bouncen
  delay(300);

  // Om te voorkomen dat de ledring gaat knipperen wachten we tot het knopje losgelaten is
  while(digitalRead(BTN_PIN)) {
    delay(100);
  }
}
```

Weergeven ledring

Omdat het een nachtlampje is willen we hem een iets gelig licht geven. Dat kan door de waarde van het blauwe licht iets lager te maken dan de recht. Die kleur kunnen we voor elke pixel als volgt instellen.

```
for (int i = 0; i < strip.numPixels(); i++) {
  strip.setPixelColor(i, 255, 255, 75);
}
```

Vervolgens willen we de helderheid van de ring aanpassen aan de stand van de potentiometer. Dus moeten we de waarde daarvan eerst uitlezen met de `analogRead()` funcie en opslaan. De helderheid van de ledring moet ingesteld worden met een waarde tussen de 0 en de 255 maar de maximale waarde die uit de `analogRead()` functie komt is 1023, dus delen we die door 4. Daarna stellen we de helderheid van de strip ermee in.

```
int brightness = analogRead(POT_PIN) / 4;
strip.setBrightness(brightness);
```

Dit alles moet natuurlijk alleen gebeuren als de ledring aan moet staan. Als hij uit moet staan dan moet de helderheid 0 zijn. Dit kan met een if statement als volgt.

```
if (lightOn) {
  // Alles om het lampje aan te zetten en de helderheid in te stellen
} else {
  // Als hij uit moet staan zetten we de felheid op 0
  strip.setBrightness(0);
}
```

Tot slot moeten we niet vergeten om de wijzingen aan de pixels naar de ledring te versturen.

```
strip.show();
```

5.2.4 Volledig script

Als je alle stappen gevolgd hebt moet je op de volgende code uitkomen.

```
// We voegen de Neopixel library toe waarmee we de ledring kunnen aansturen
#include <Adafruit_NeoPixel.h>

// We definiëren aan welke pinnen we het knopje, de potentiometer en ledring aangesloten hebben
#define POT_PIN A0
#define BTN_PIN D8
#define LED_PIN D4

// Onze ledring heeft 60 ledjes, dit slaan we op en we maken een ledstrip object aan
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

// We maken een globale variabele aan waarin we onthouden of het nachlampje aan of uit staat
boolean lightOn = true;

void setup() {
  // We starten de ledstrip op en zetten hem aan
  strip.begin();
  strip.show();

  // We definiëren dat het knopje als input gebruik moet worden
  pinMode(BTN_PIN, INPUT);
}

void loop() {
  // Als het knopje ingedrukt wordt moet het lampje aan of uit gaan
  if (digitalRead(BTN_PIN)) {
    // We veranderen de waarde naar het tegenovergestelde van zijn vorige waarde, aan wordt dus uit en uit wordt aan
    lightOn = !lightOn;

    // We wachten heel even om rekening te houden met bouncen
    delay(300);

    // Om te voorkomen dat de ledring gaat knipperen wachten we tot het knopje losgelaten is
    while(digitalRead(BTN_PIN)) {
      delay(100);
    }
  }

  // We bepalen of het nachtlampje aan moet staan
  if (lightOn) {
    // Zo ja dan stellen we eerst de kleur in van elke pixel
    for (int i = 0; i < strip.numPixels(); i++) {
      strip.setPixelColor(i, 255, 255, 75);
    }

    // Vervolgens bepalen we hoe fel hij moet zijn door de waarde van de potmeter uit te lezen
    // Dit is een waarde tussen de 0 en 1024, de brightness moet ingesteld worden tussen 0 en 255, dus delen he het door 4
    int brightness = analogRead(POT_PIN) / 4;

    // We stellen de felheid van de ledring in
    strip.setBrightness(brightness);
  } else {
    // Als hij uit moet staan zetten we de felheid op 0
    strip.setBrightness(0);
  }

  // We sturen alle instellingen die we gedaan hebben naar de ledring om weergeven te worden
  strip.show();
}
```

5.3 Wakeuplight

De code voor een wakeuplight lijkt veel op die voor het nachtlampje met een paar aanpassingen. Allereerst zullen we met wifi verbinden en de huidige tijd ophalen. Vervolgens zullen we op een vooraf ingestelde tijd het lampje aanzetten.

5.3.1 Voorbereiding

Allereerst voegen we de libraries toe om met wifi te verbinden en de NTPclient om de tijd op te halen. Vergeet deze niet te installeren.

```
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
```

We maken twee constante variabelen aan waarin we het uur en de minuut waarop de lamp aan moet gaan in opslaan. Daarna slaan we de de naam van het wifi netwerk en het wachtwoord in.

```
// We stellen in hoe laat de lamp aan moet gaan
const int wakeupHour = 8;
const int wakeupMinute = 30;

// We stellen het wifi netwerk en wachtwoord in
const char *ssid     = "netwerknaam";
const char *password = "wachtwoord";
```

Tot slot maken we een wifiverbinding object en een ntpclient object aan.

```
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");
```

5.3.2 Setup

We moeten eerst een verbinding met wifi maken.

```
// We maken verbinding met het wifi netwerk
WiFi.begin(ssid, password);
```

Omdat het fijn is om te weten of dat lukt en of de ESP niet vastgelopen is kun je met de volgende code op de ledring laten zien dat we nog steeds bezig zijn met het maken van een verbinding.

```
// Zolang we nog niet met het wifi verbonden zijn updaten we een pixel van de ring naar oranje
int status = 0;
while (WiFi.status() != WL_CONNECTED) {
  delay(500);

  if (status < LED_COUNT) {
    strip.setPixelColor(status++, 255, 255, 0);
    strip.show();
  }
}
```

Zodra we een wifi verbinding hebben maken we verbinding met een NTP server om de tijd op te halen. Ook stellen we de tijdszone in op europese zomertijd. Voor wintertijd kun je 3600 als offset gebruiken.

```
// We starten de NTPclient op en stellen de tijdszone in op GMT+2 (europese zomertijd)
timeClient.begin();
timeClient.setTimeOffset(7200);
```

5.3.3 Loop

Elke loop vragen we de huidige tijd op.

```
timeClient.update();
```

Als de huidige tijd overeen komt met de wekkertijd dan zetten we de lamp aan.

```
if (timeClient.getHours() == wakeupHour && timeClient.getMinutes() == wakeupMinute) {
  lightOn = true;
}
```

5.3.4 Verbeter mogelijkheden

Er zijn nog wat mogelijkheden om de functionaliteit van de wakeuplight te verbeteren. Zo zou je een aantal minuten voor de wekkertijd de lamp langzaam aan laten gaan. Ook zou je een snooze modus in kunnen bouwen of een manier waarmee je met de draaiknop en het knopje de wekkertijd in kunt stellen.

5.3.5 Volledig script

Als je de stappen gevolgd hebt dan kom je op de volgende code uit.

```
// We voegen de Neopixel library toe waarmee we de ledring kunnen aansturen
#include <Adafruit_NeoPixel.h>

// We voegen de Wifi en de NTPclient libraries toe waarmee we met wifi kunnen verbinden en de tijd kunnen ophalen
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>

// We stellen in hoe laat de lamp aan moet gaan
const int wakeupHour = 8;
const int wakeupMinute = 30;

// We stellen het wifi netwerk en wachtwoord in
const char *ssid     = "netwerknaam";
const char *password = "wachtwoord";

// We maken een Wifi en NTPclient object aan
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");

// We definiëren aan welke pinnen we het knopje, de potentiometer en ledring aangesloten hebben
#define POT_PIN A0
#define BTN_PIN D8
#define LED_PIN D4

// Onze ledring heeft 60 ledjes, dit slaan we op en we maken een ledstrip object aan
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

// We maken een globale variabele aan waarin we onthouden of het nachlampje aan of uit staat
boolean lightOn = false;

void setup() {
  // We starten de ledstrip op en zetten hem aan
  strip.begin();

  // We maken de ledstrip rood om aan te geven dat we nog niet met wifi verbonden zijn
  for (int i = 0; i < strip.numPixels(); i++) {
    strip.setPixelColor(i, 255, 0, 0);
  }
  strip.setBrightness(25);
  strip.show();

  // We maken verbinding met het wifi netwerk
  WiFi.begin(ssid, password);

  // Zolang we nog niet met het wifi verbonden zijn updaten we een pixel van de ring naar oranje
  int status = 0;
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);

    if (status < LED_COUNT) {
      strip.setPixelColor(status++, 255, 255, 0);
      strip.show();
    }
  }

  // We starten de NTPclient op en stellen de tijdszone in op GMT+2 (europese zomertijd)
  timeClient.begin();
  timeClient.setTimeOffset(7200);

  // We definiëren dat het knopje als input gebruik moet worden
  pinMode(BTN_PIN, INPUT);

  // We maken de ledstrip groen om aan te geven dat we succesvol opgestart zijn
  for (int i = 0; i < strip.numPixels(); i++) {
    strip.setPixelColor(i, 0, 255, 0);
  }
  strip.show();
  delay(500);
}

void loop() {
  // We updaten de NTPclient met de huidige tijd
  timeClient.update();

  // Als de huidige tijd overeen komt met de wekker tijd moet de lamp aan
  if (timeClient.getHours() == wakeupHour && timeClient.getMinutes() == wakeupMinute) {
    lightOn = true;
  }

  // Als het knopje ingedrukt wordt moet het lampje aan of uit gaan
  if (digitalRead(BTN_PIN)) {
    // We veranderen de waarde naar het tegenovergestelde van zijn vorige waarde, aan wordt dus uit en uit wordt aan
    lightOn = !lightOn;

    // We wachten heel even om rekening te houden met bouncen
    delay(300);

    // Om te voorkomen dat de ledring gaat knipperen wachten we tot het knopje losgelaten is
    while(digitalRead(BTN_PIN)) {
      delay(100);
    }
  }

  // We bepalen of het nachtlampje aan moet staan
  if (lightOn) {
    // Zo ja dan stellen we eerst de kleur in van elke pixel
    for (int i = 0; i < strip.numPixels(); i++) {
      strip.setPixelColor(i, 255, 255, 75);
    }

    // Vervolgens bepalen we hoe fel hij moet zijn door de waarde van de potmeter uit te lezen
    // Dit is een waarde tussen de 0 en 1024, de brightness moet ingesteld worden tussen 0 en 255, dus delen he het door 4
```

5.4 Klok

Dit project bouwt voort op de code van het nachtlampje en de wakeuplight. Het verschil zit vooral in de loop functie waarin we de juiste pixels aan zetten om de tijd te laten zien. Het mooie is dat de ring precies 60 leds heeft, wat een toeval.

5.4.1 Loop

We berekenen welke pixel er aan moet voor het uur, minuut en seconde.

```
int minuutPixel = timeClient.getMinutes();
int uurPixel = (timeClient.getHours() % 12) * 5;
int secondePixel = timeClient.getSeconds();
```

Vervolgens loopen we door alle pixels heen. Op de 12, 3, 6 en 9 maken we een pixel wit tenzij een van de wijzers er staat. De minuten geven we aan met rood, het uur met groen en de seconde met blauw.

```
for (int i = 0; i < strip.numPixels(); i++) {
  int red = (i == minuutPixel) ? 255 : 0;
  int green = (i == uurPixel) ? 255 : 0;
  int blue = (i == secondePixel) ? 255 : 0;

  if (i % 15 == 0 && !red && !green && !blue) {
    red = 255;
    green = 255;
    blue = 255;
  }

  strip.setPixelColor(LED_COUNT - 1 - i, red, green, blue);
}
```

Met wat creativiteit kun je ook hele andere animaties maken voor de klok. Bedenk wat leuks!

5.4.2 Volledig script

Het volledige klok script is als volgt.

```
#include <Arduino.h>
// We voegen de Neopixel library toe waarmee we de ledring kunnen aansturen
#include <Adafruit_NeoPixel.h>

// We voegen de Wifi en de NTPclient libraries toe waarmee we met wifi kunnen verbinden en de tijd kunnen ophalen
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>

// We stellen het wifi netwerk en wachtwoord in
const char *ssid     = "netwerknaam";
const char *password = "wachtwoord";

// We maken een Wifi en NTPclient object aan
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");

// We definiëren aan welke pinnen we het knopje, de potentiometer en ledring aangesloten hebben
#define POT_PIN A0
#define BTN_PIN D8
#define LED_PIN D4

// Onze ledring heeft 60 ledjes, dit slaan we op en we maken een ledstrip object aan
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  // We starten de ledstrip op en zetten hem aan
  strip.begin();

  // We maken de ledstrip rood om aan te geven dat we nog niet met wifi verbonden zijn
  for (int i = 0; i < strip.numPixels(); i++) {
    strip.setPixelColor(i, 255, 0, 0);
  }
  strip.setBrightness(25);
  strip.show();

  // We maken verbinding met het wifi netwerk
  WiFi.begin(ssid, password);

  // Zolang we nog niet met het wifi verbonden zijn updaten we een pixel van de ring naar oranje
  int status = 0;
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);

    if (status < LED_COUNT) {
      strip.setPixelColor(status++, 255, 255, 0);
      strip.show();
    }
  }

  // We starten de NTPclient op en stellen de tijdszone in op GMT+2 (europese zomertijd)
  timeClient.begin();
  timeClient.setTimeOffset(7200);

  // We maken de ledstrip blauw om aan te geven dat we succesvol opgestart zijn
  for (int i = 0; i < strip.numPixels(); i++) {
    strip.setPixelColor(i, 0, 0, 255);
  }
  strip.show();
  delay(500);
}

void loop() {
  // We updaten de NTPclient met de huidige tijd
  timeClient.update();
  int minuutPixel = timeClient.getMinutes();
  int uurPixel = (timeClient.getHours() % 12) * 5;
  int secondePixel = timeClient.getSeconds();

  for (int i = 0; i < strip.numPixels(); i++) {
    int red = (i == minuutPixel) ? 255 : 0;
    int green = (i == uurPixel) ? 255 : 0;
    int blue = (i == secondePixel) ? 255 : 0;

    if (i % 15 == 0 && !red && !green && !blue) {
      red = 255;
      green = 255;
      blue = 255;
    }

    strip.setPixelColor(LED_COUNT - 1 - i, red, green, blue);
  }

  // Vervolgens bepalen we hoe fel hij moet zijn door de waarde van de potmeter uit te lezen
  // Dit is een waarde tussen de 0 en 1024, de brightness moet ingesteld worden tussen 0 en 255, dus delen he het door 4
  int brightness = analogRead(POT_PIN) / 4;

  // We stellen de felheid van de ledring in
  strip.setBrightness(brightness);

  // We sturen alle instellingen die we gedaan hebben naar de ledring om weergeven te worden
  strip.show();

  delay(500);
}
```

5.5 Agenda

In dit project gaan we de tijd van de eerstvolgende google agenda afspraak ophalen en een balk laten aftellen totdat de afspraak begint. Allereest zullen we een Google script schrijven dat je agenda uitleest en een lijstje met afspraken voor de komende 24 uur opstelt. De ESP kan vervolgens via een url dat lijstje opvragen en de balk weergeven. De enige hobbel in het opvragen van de data is dat er een redirect in zit met een tweede url voordat we de echte data kunnen opvragen.

5.5.1 Google script

Volg de volgende stappen om het google script aan te maken.

Script aanmaken

Navigeer naar https://script.google.com, log in met je google account waar ook je agenda aan verbonden is en maak een nieuw project aan door op de `Nieuw project` knop te drukken.

Script schrijven

In de editor kunnen we vervolgens een script schrijven. Allereerst moeten we verbinding maken met de agenda. Hiervoor moet je de naam van de kalender invullen, vaak is dit je gmail email adres.

```
var cal = CalendarApp.getCalendarById('calendar-name');
if (cal == undefined) {
  return ContentService.createTextOutput("no access to calendar");
}
```

Vervolgens berekenen we de start en eindtijd waartussen we alle agenda afspraken willen ophalen. In dit geval een periode van 24 uur vanaf nu.

```
var start = new Date();
const oneday = 24*3600000; // [msec]
const stop = new Date(start.getTime() + 7 * oneday);
```

We halen alle events in deze periode uit de agenda op.

```
var events = cal.getEvents(start, stop);
```

Daarna maken we een string aan waaraan we alle agenda afspraken van onszelf of waarvan we aangegeven hebben dat we aanwezig zijn toevoegen. We voegen aan die string de startijd in miliseconden epoch, de starttijd in tekst, of het event de hele dag duurt en de titel toe van elkaar gescheiden met een tab. Is dat wat veel als we eigenlijk alleen de tijd in epoch van de eerstvolgende afspraak nodig hebben? Ja eigenlijk wel, maar dit is een demo, het werkt, en nu heb je meer kennis om er ook andere dingen mee te doen.

```
var str = '';
for (var ii = 0; ii < events.length; ii++) {

  var event=events[ii];
  var myStatus = event.getMyStatus();

  switch(myStatus) {
    case CalendarApp.GuestStatus.OWNER:
    case CalendarApp.GuestStatus.YES:
    case CalendarApp.GuestStatus.MAYBE:
      str += event.getStartTime().getTime() + '\t' +
             event.getStartTime() + '\t' +
             event.isAllDayEvent() + '\t' +
             event.getTitle() +'\n';
      break;
    default:
      break;
  }
}
return ContentService.createTextOutput(str);
```

Script implementeren

Rechtsboven zie je als het goed is een blauwe "implementeren" knop. Die moeten we gebruiken om een link voor het script te genereren. Maak een nieuwe implementatie aan. Kies als type een `web-app`, voer hem uit als jezelf en kies bij toegang voor iedereen. Dit betekend technisch gezien dat iedereen met de url je agenda kan zien. Nu is de url heel erg moeilijk te gokken, maar denk er wel even over na. Als ik beveiliging moet gaan uitleggen gaan we allemaal huilen, iets voor een andere keer.

Noteer van je implementatie de Implementatie-ID. Die hebben we straks nodig om de data op te halen.

Volledig script

Voor de volledigheid hier het hele script.

```
function doGet(e) {
  var cal = CalendarApp.getCalendarById('calendar-name');
  if (cal == undefined) {
    return ContentService.createTextOutput("no access to calendar");
  }

  var start = new Date();
  const oneday = 24*3600000; // [msec]
  const stop = new Date(start.getTime() + 7 * oneday);
  Logger.log(start);
  Logger.log(stop);

  var events = cal.getEvents(start, stop);

  var str = '';
  for (var ii = 0; ii < events.length; ii++) {

    var event=events[ii];
    var myStatus = event.getMyStatus();

    switch(myStatus) {
      case CalendarApp.GuestStatus.OWNER:
      case CalendarApp.GuestStatus.YES:
      case CalendarApp.GuestStatus.MAYBE:
        str += event.getStartTime().getTime() + '\t' +
               event.getStartTime() + '\t' +
               event.isAllDayEvent() + '\t' +
               event.getTitle() +'\n';
        break;
      default:
        break;
    }
  }
  return ContentService.createTextOutput(str);
}
```

5.5.2 ESP script

Het ESP script zal uit een aantal onderdelen bestaan. Allereerst een functie waarmee we een HTTPS GET request naar een server kunnen doen en de reactie van de server terug krijgen. Deze functie zullen we gebruiken in een andere functie waarmee we de tijd van de eerstvolgende afspraak ophalen. Deze tijd zullen we vervolgens opslaan en gebruiken voor het weergeven van een aftel ring.

HTTPS GET request

Voor een GET request hebben we twee dingen nodig, de host oftewel het domein van de server en de url die we willen opvragen. Het resultaat van een GET request is een string met het bericht. We maken dus een functie aan waar we de host en de url aan geven en deze zal het bericht als String teruggeven.

```
String httpsGet(String host, String url) {
  // Code komt hier
}
```

In de functie maken we een `WiFiClientSecure` object aan waarmee we de request kunnen uitvoeren. Normaliter zou je nu ook een methode toevoegen waarmee je de identiteit van de server waarmee je verbint kunt controleren. Dat is onderdeel van een beveiligde verbinding. Voor deze keer laten we dat achterwege en zetten we een onbeveiligde verbinding op.

```
WiFiClientSecure client;
client.setInsecure();
```

We maken verbinding met de host op de https poort. Als dat niet lukt geven we een lege string terug.

```
const int httpPort = 443; // 80 is voor HTTP / 443 is voor HTTPS
if (!client.connect(host, httpPort)) { //works!
  return "";
}
```

Hierna sturen we de GET request naar de server.

```
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
```

We wachten totdat we een header ontvangen.

```
while (client.connected()) {
  String line = client.readStringUntil('\n');
  if (line == "\r") {
    break;
  }
}
```

Als er daarna een bericht terug komt slaan we dat op in een string.

```
String response = "";
while (client.available()) {
  char c = client.read();
  response.concat(c);
}
```

Deze string geven we terug.

```
return response;
```

Eerstvolgende afspraak ophalen

We hebben een prachtige URL gegenereerd voor het ophalen van de agenda afspraken via een google script. De enige hindernis die we hebben is dat als we de url benaderen van het google script deze een redirect zal teruggeven. Je krijgt bij het opvragen van de url dus niet meteen de data die je wilt hebben maar een nieuwe url. Bij die nieuwe url kun je vervolgens wel de data ophalen. We moeten in het script dus inbouwen dat we eerst de redirect url verkrijgen en daarna de data opvragen.

Dit alles zal uitgevoerd worden door de `getNextAppointmentTime()` functie die een integer met de tijd in epoch van de volgende afspraak terug zal geven.

REDIRECT URL OPHALEN

Allereerst halen we de redirect url op van het google script en die slaan we op in een string.

```
String redirectMessage = httpsGet(host, url);
```

We hebben alleen de url nodig, dus die moeten we uit het bericht filteren. De url begint met `/macros` en eindigt voor `>here` dus we zoeken op waar dat in de string staat. Daarmee bepalen we de redirect url.

```
int from = redirectMessage.indexOf("/macros");
int to = redirectMessage.indexOf("\">here");
String redirectUrl = redirectMessage.substring(from, to);
```

In een url wordt soms de `&` vervangen voor een `&`. Dat geeft problemen als we de url later willen gaan gebruiken. Dus deze moeten we weer terug vervangen. Dat doen we als volgt.

```
redirectUrl.replace("&amp;", "&");
```

TIJD OPHALEN

Nu we de redirect url weten kunnen we de tijd van de eerstvolgende afspraak ophalen.

```
String calenderItems = httpsGet(redirectHost, redirectUrl);
```

In de response staat veel meer informatie dan dat we nodig hebben. De tijd staat na de eerste enter (dat is een `\n` in ascii) en voor de eerste tab (dat is een `\t` in ascii). Dus de plek daarvan zoeken we op.

```
int start = calenderItems.indexOf("\n");
int end = calenderItems.indexOf("\t");
```

Vervolgens halen we het nummer uit de string. Het google script geeft de epoch tijd in milliseconden maar wij werken in dit script met seconden. Dus laten we de laatste 3 cijfers van het getal gewoon achterwege. De string zetten we vervolgens om in een nummer en geven we terug.

```
String epoch = calenderItems.substring(start+1, end-3);
return epoch.toInt();
```

Ring animeren

Om de ring te animeren volgen we een aantal stappen. Eerst kijken we of we al weten wanneer de volgende afspraak is. Zo niet dan moeten we die ophalen. Ook willen we elke 10 minuten controleren of de agenda toevallig niet aangepast is. De tijd van de eerstvolgende afspraak slaan we op in een variabele.

```
if (nextAppointmentTime == 0 || timeClient.getMinutes() % 10 == 0) {
  nextAppointmentTime = getNextAppointmentTime();
}
```

Hierna berekenen we hoe lang het nog duurt tot de volgende afspraak.

```
int minutesToAppointment = (nextAppointmentTime - timeClient.getEpochTime()) / 60;
```

Als het minder dan 60 minuten is tot de volgende afspraak willen we het aantal pixels aan zetten overeenkomstig met het aantal minuten dat het nog duurt.

```
for (int i = 0; i < strip.numPixels(); i++) {
  if (i < minutesToAppointment && minutesToAppointment < 60) {
    strip.setPixelColor(i, 0, 255, 0);
  } else {
    strip.setPixelColor(i, 0, 0, 0);
  }
}
strip.show();
```

Als de tijd van de afspraak bereikt is dan moeten we de tijd van de eerstvolgende afspraak ophalen.

```
if (minutesToAppointment <= 0) {
  nextAppointmentTime = getNextAppointmentTime();
}
```

Vervolgens wachten we een minuut om de ring weer te updaten.

Volledig script

```
#include <Arduino.h>

// We voegen de Wifi en de NTPclient libraries toe waarmee we met wifi kunnen verbinden en de tijd kunnen ophalen
#include <ESP8266WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>

// We voegen de Neopixel library toe waarmee we de ledring kunnen aansturen
#include <Adafruit_NeoPixel.h>

// We stellen het wifi netwerk en wachtwoord in
const char* ssid     = "SSID";
const char* password = "PASSWORD";

const char* host = "script.google.com";
const char* redirectHost = "script.googleusercontent.com";
const char* url = "/macros/s/YOUR_APP_ID/exec";

// We maken een Wifi en NTPclient object aan
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");

// We definiëren aan welke pinnen we de ledring aangesloten hebben en we maken een ledstrip object aan
#define LED_PIN D4
#define LED_COUNT 60
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

// We maken een variabele aan waarin we de tijd van de volgende afspraak aanmaken.
int nextAppointmentTime = 0;

/**
 * Functie om door middel van een HTTPS GET request van een host en url data op te vragen
 * @param host Het domein dat benaderd moet worden
 * @param url De url op het domein dat benaderd moet worden
 * @return Een string met de volledige response
 */
String httpsGet(String host, String url) {
  // We gebruiken de WiFiClientSecure klasse om een TCP verbinding op te zetten
  WiFiClientSecure client;

  // Dit is de magische regel waarmee we alle beveiliging negeren en waardoor alles werkt
  // Als je een echt veilige verbinding op wilt zetten zou je nu dingen met certificaten moeten doen
  // Dat is kei veel werk en voor een simpele agenda export laten we het achterwege
  client.setInsecure();

  // We maken verbinding met de server
  const int httpPort = 443; // 80 is voor HTTP / 443 is voor HTTPS
  if (!client.connect(host, httpPort)) { //works!
    Serial.println("connection failed");
    return "";
  }

  Serial.print("Requesting URL: ");
  Serial.println(url);

  // We sturen de GET request naar de server
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");

  // We wachten totdat we een header ontvangen
  while (client.connected()) {
    String line = client.readStringUntil('\n');
    if (line == "\r") {
      Serial.println("headers received");
      break;
    }
  }

  // Als er daarna een bericht terug komt slaan we het op in een string
  String response = "";
  while (client.available()) {
    char c = client.read();
    Serial.write(c);
    response.concat(c);
  }

  Serial.println();
  Serial.println("closing connection");

  // We geven de string met het bericht terug
  return response;
}

/**
 * Functie om de tijd in epoch van de volgende afspraak op te halen
 * @returns Een integer met de tijd in epoch van de volgende afspraak
 */
int getNextAppointmentTime() {
  Serial.print("connecting to ");
  Serial.println(host);

  // We halen de redirect url op van het google script
  String redirectMessage = httpsGet(host, url);

  Serial.println("Redirect message: ");
  Serial.println(redirectMessage);

  // Er zit wat HTML in het redirect bericht en we hebben alleen de url nodig
  // Daarom bepalen we hier waar deze precies staat
  int from = redirectMessage.indexOf("/macros");
  int to = redirectMessage.indexOf("\">here");

  // Vervolgens slaan we de url op
```

5.5.3 Verbeter mogelijkheden

Een optie is om meerdere agenda's uit te lezen en de kleur van die agenda mee te sturen. Dan kun je de kleur van de ledring aanpassen zodat je weet voor welke agenda de afspraak is. Verder zou je ook de kleur van groen naar rood kunnen laten verlopen hoe dichter je bij de tijd van de afspraak ben. Een andere optie is om de agenda afspraken, optioneel in kleuren, over een klok heen te leggen. Zodat je in een oogopslag kunt zien wanneer je bepaalde afspraken hebt. Stel dat je je collegerooster inlaad, dan zou je met rode strepen op de klok je collegetijden aan kunnen geven. Dan weet je 's ochtends meteen wat je te wachten staat die dag.

6. Probleemoplosser

Help, alles gaat mis! Gelukkig hebben we hier een aantal tips om je problemen op te lossen. Het is natuurlijk altijd mogelijk om iemand om hulp te vragen. Voor alle problemen hier niet beschreven raden we aan om het volgende liedje op vol volume af te spelen: https://youtu.be/oeCVXHPZ9WY.

6.1 Software

6.1.1 Ik pas de kleur van de pixels aan maar ik zie de ledring niet veranderen

Als je de `setPixelColor()` functie of de `setBrightness()` gebruikt in de Neopixel library zul je eerst de `show()` functie aan moeten roepen voordat je wijzigingen op de ledring getoond worden.

6.1.2 Ik krijg een error dat ik niet naar de COM poort kan schrijven of dat deze bezet is

Onder Windows mag maar één programma tegelijkertijd met een COM poort communiceren. Als je dus een seriële monitor open hebt staan en tegelijkertijd de chip probeert te programmeren werkt dat niet. Controleer of er niet per ongeluk nog een verbinding open staat.

Mogelijk zijn er twee apparaten met hetzelfde COM poort nummer. Dit is een uitzonderlijk geval en heeft vaak te maken met verbonden bluetooth apparaten. Probeer een andere USB poort te gebruiken en controleer of je de goede poort geselecteerd hebt.

Het kan gebeuren dat er wat mis gaat en de COM poort in Windows gereserveerd blijft maar dat het programma dat deze reservering plaatste al is afgesloten of is gechashed. In dat geval is de enige oplossing om je computer opnieuw op te starten.

6.1.3 Ik zie alleen maar rare tekens in mijn seriële monitor

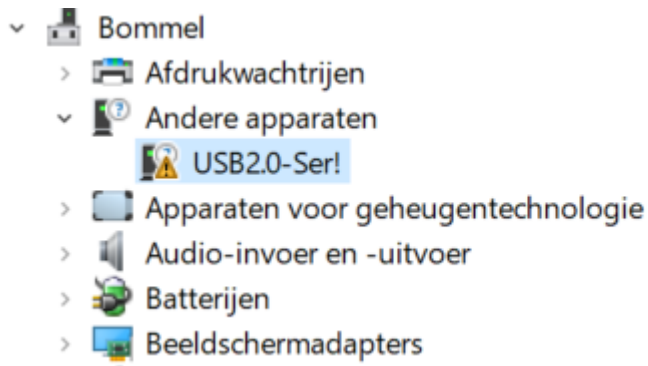
Als je seriële monitor alleen maar rare tekens laat zien is waarschijnlijk de bautrate van je monitor niet hetzelfde als dat van je microcontroller. Zoek in je code naar welke waarde dit is, bijvoorbeeld `Serial.begin(9600);` stelt de bautrate in op 9600. Zet je monitor dus op dezelfde waarde.

6.1.4 Er is geen COM poort beschikbaar van de ESP

Wanneer er geen COM poort verschijnt voor de ESP als je deze met de USB kabel verbint, of het knopje voor COM poorten blijft grijs kan het zijn dat de driver niet goed geïnstalleerd is. Dit is alleen een probleem op Windows.

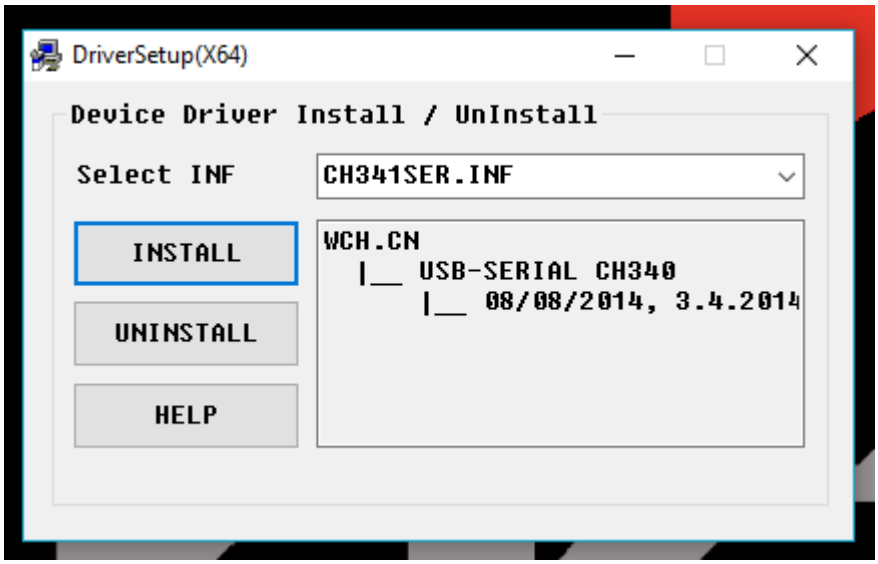
Controleren of de driver niet goed geïnstalleerd is

Om te controleren of de driver niet goed geïnstalleerd is moeten we de Device Manager (Apparaatbeheer) openen. Druk op `⊞ Win` + `X` en vervolgens op `Device Manager` of `Apparaatbeheer`. Kijk in de lijst met apparaten of je een apparaat genaamd `USB2.0-Ser!` ziet staan. Zo ja, dan zijn je drivers niet goed geïnstalleerd.



Correcte drivers installeren

De seriële communicate op de ESP die we jullie geleverd hebben wordt verzorgt door een CH340 chip. Hiervoor moet een correcte driver worden geïnstalleerd. Deze kun je downloaden via https://learn.sparkfun.com/tutorials/how-to-install-ch340-drivers#drivers-if-you-need-them. Download de Windows (EXE) Driver executable en voer hem uit. Druk eerst op de `Uninstall` knop en vervolgens op de `Install` knop. Als het goed is moet je ESP nu in de Device Manager herkent worden als een COM device en een poort nummer krijgen.



6.2 Hardware

6.2.1 Als ik mijn ledring felle kleuren of veel wit laat weergeven gaat hij uit

Hoe feller en witter de kleuren, hoe meer stroom de ledring gebruikt. Als de ledring volledig wit is op volle helderheid gebruikt hij ongeveer 2A. Het kan zijn dat de voeding die je gebruikt dit niet aan kan, kijk of je de kleuren minder fel kunt maken of zet je brightness iets lager.

6.2.2 Er komt rook uit mijn microcontroller!

Trek snel de USB kabel er uit! Je hebt zojuist de magische rook laten ontsnappen. Zodra de magische rook, die alle elektronica laat werken, ontsnapt is kan het niet meer gerepareerd worden. De enige oplossing is huilen in een hoekje.