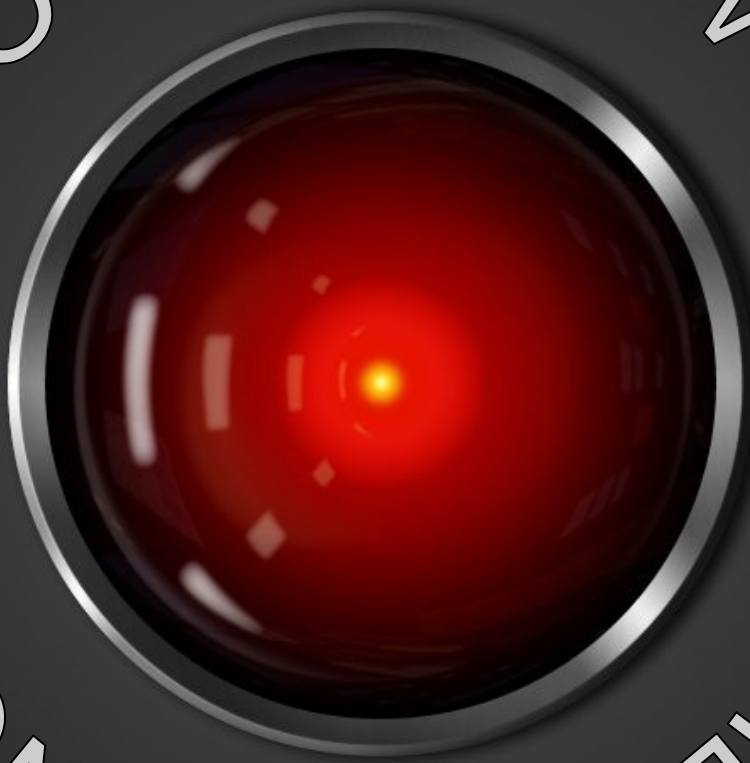
INTERACTIEF ONTWERPEN MET ARDUINO



WAT IS WAT?

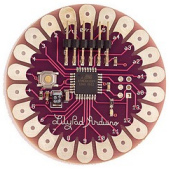
INTERACTIEF ONTWERPEN: GAAT OVER DE WISSELWERKING TUSSEN MENSEN EN MACHINES, TUSSEN DE FYSIEKE EN DE VIRTUELE WERELD.

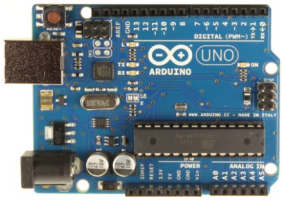
ARDUINO: OPEN-SOURCE MICROCONTROLER PLATVORM (BESTAANDE UIT HARDWARE EN SOFTWARE). VOORAL ONTWIKKELD VOOR MENSEN MET EEN NIET TECHNISCHE ACHTERGROND ALS KUNSTENAARS EN VORMGEVERS.

MICROCONTROLER: EEN KLEINE COMPUTER OP ÉÉN PRINTPLAAT WAAR-OP SENSOREN (KUNNEN IETS WAARNEMEN) EN ACTUATOREN (KUNNEN IETS DOEN) KUNNEN WORDEN AANGESLOTEN.

OPEN SOURCE: DE BRONCODE IS VRIJ BESCHIKBAAR EN DE HARD EN SOFTWARE KAN DOOR IEDEREEN GEKOPIEERD AANGEPAST EN VER-SPREID WORDEN.

IDE: INTEGRATED DEVELOPMENT ENVIRONMENT. COMPUTERSOFTWARE OM COMPUTERSOFTWARE TE MAKEN. IN ONS GEVAL DE ARDUINO SOFTWARE.





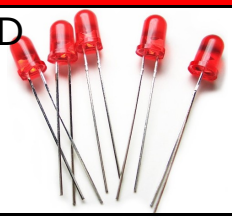
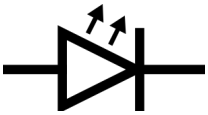
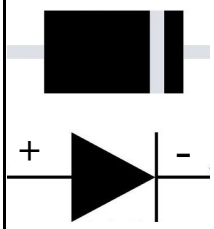
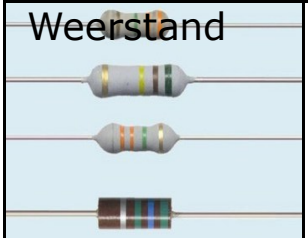
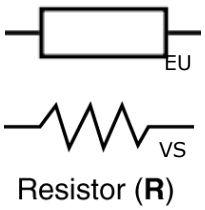
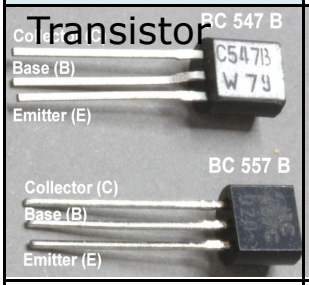
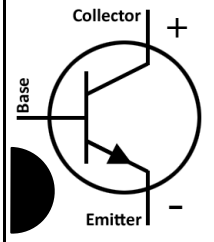
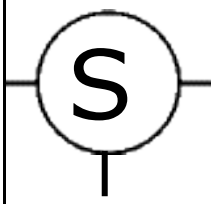
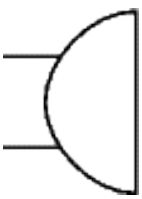
Meer informatie:

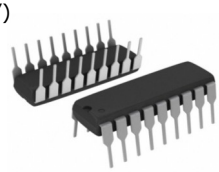
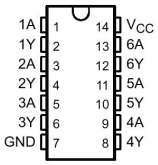
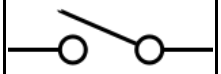
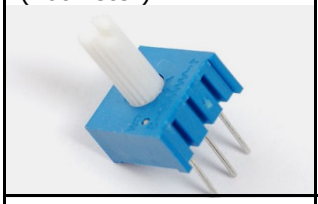
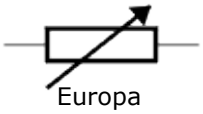
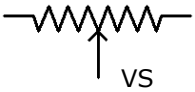
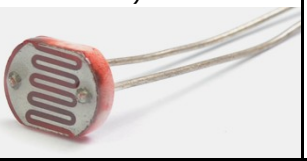
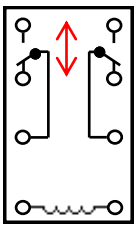
Er is op internet veel informatie te vinden over Arduino en Arduino projecten.
* Op de Arduino site zelf staat zeer veel informatie.
http://www.arduino.cc/
* Op ladyada.net staat onder meer een goede tutorial.
http://www.ladyada.net/learn/arduino/
* Van Earthshine Electronics een starter kit met veel programma uitleg.
eBook en sketches hier te downloaden.
http://www.earthshineelectronics.com/10-arduino-starter-kit.html
* Van Jeremy Blum een goede youtube tutorial.
http://www.jeremyblum.com/category/arduino-tutorials/page/4/

Inhoud

Elektronische Componenten.

Hieronder een lijstje met de belangrijkste componenten die we gebruiken in onze experimenteer kit. Je kan hier snel bekijken hoe ze eruit zien en hoe ze werken. De meeste componenten hebben ook een symbool, dat staat helemaal rechts.

Component	Wat doet het?	Waar let je op?	Symbool
LED 	Als er een klein beetje stroom doorheen loopt geeft de led licht. De stroom kan er maar in een richting door, net als bij een diode. Vandaar de naam led: light emitting diode.	-Altijd aansluiten met een weerstand (bv 330 ohm), anders gaat hij stuk. -Het lange pootje moet aan de plus kant, het korte aan de min.	
Diode 	Een diode laat de stroom maar in één richting door. Het is meestal een cilindertje waaruit twee draden komen. Aan een kant is een ring waaraan je kan zien aan welke kant de plus en min komen.	Een diode kan je snel verkeerd aansluiten en dan werkt je circuit niet of er kan iets kapot gaan. De ring moet aan de min kant worden aangesloten.	
Weerstand 	Een weerstand vermindert de stroom in een circuit zodat alle onderdelen de juiste hoeveelheid stroom krijgen. De waarde van een weerstand wordt uitgedrukt in ohm (spreek uit: oom).	Ze zijn klein en moeilijk van elkaar te onderscheiden. Aan de gekleurde ringen kan je de waarde aflezen (zie tabel op de volgende pagina). Je kan ook meten met de multimeter.	
Transistor 	Een transistor gebruikt een klein stroompje om een veel grotere stroom te schakelen of te versterken. Er gaat een stroom (of meer) stroom lopen als er een kleine stroom door de basis gaat (het middelste pootje).	Zorg dat het collector pootje aan de pluskant is en het emitter pootje aan de min kant. Transistors zijn er in veel maten en soorten. Wij gebruiken meestal de 547, 557 of de 2222AG.	
Servo motor 	Een servo is een motor die een elektrische impuls vertaalt in een bepaalde positie. Normaal kan de motor maar 180 graden draaien. Alleen 'gemodificeerde' servo's kunnen blijven draaien.	De servo heeft drie aansluitdraden. Zwart voor min, rood plus en de derde is voor de puls en gaat naar een pin van de arduino . Grotere servo's met een batterij aansluiten.	
DC Motor 	Een DC (=gelijkstroom) motor kan je aansluiten op een batterij. Afhankelijk van de spanning kan de motor sneller of langzamer draaien. Als je de plus en min aansluiting verwisselt draait hij andersom.	De motor gebruikt teveel stroom om rechtstreeks op een pin van de arduino aan te sluiten. Dit kan alleen met behulp van een relais of een transistor.	
Piezo element 	In een piezo element worden kristallen vervormt door elektriciteit. Een stroom van elektrische pulsen geeft een toon. De behuizing kan ook alleen een metalen schijfje zijn.	De plus (rood) en min (zwart) moeten goed worden aangesloten. Een piezo element kan ook als geluidsensor dienen. De vervorming van de kristallen wekken dan elektriciteit op.	

Component	Wat doet het?	Waar let je op?	Symbool
IC (Integrated Circuit) ('Chip') 	Er zijn heel veel verschillende soorten en maten. In de behuizing zitten complete elektronische circuits. In een ic kunnen enkele tot miljoenen (zoals bij de microprocessoren) onderdelen zitten.	De ic's die wij gebruiken hebben aan één kant een markering om de pinnen te herkennen. Ze blijven niet altijd goed zitten in een breadboard, gebruik desnoods een wasknijper.	 (Bij voorbeeld)
Drukknop 	De drukknop die wij gebruiken maakt contact als er op gedrukt wordt. Er zijn er ook die net andersom werken. De pootjes passen min of meer in een breadboard (voorzichtig induwen).	Kijk goed welke contacten worden doorverbonden bij indrukken. Meet het met de multimeter of controleer met batterij en lampje. Als hij niet werkt moet je hem misschien 90° draaien.	
Potentiometer (Potmeter) 	Een potentiometer is een variabele weerstand. Door aan de middelste pen te draaien verandert de weerstand. Onze potmeter past in een breadboard en is 10k (10 kilo ohm) maar ze bestaan in alle soorten en maten.	De potmeter heeft drie contacten. Tussen de twee buitenste is het weerstandsmateriaal. Het middelste contact is verbonden met de draaibare loper. Deze wordt verbonden met een pin van de arduino.	 Europa  VS
LDR (Lichtgevoelige weerstand) 	Een LDR (light dependent resistor) is een weerstand waarvan de waarde afhangt van de hoeveelheid licht die erop valt. In het donker is de weestand heel hoog, valt er veel licht op dan is de weerstand heel laag.	Het maakt niet uit aan welke kant je de plus of min zet. Omdat we met een arduino alleen spanning kunnen registreren en geen weerstand gebruiken we een 'spanningsdeler'.	
Relais 	Een relais is een mechanische schakelaar die bediend wordt door elektrische stroom. Met relais kunnen we grote stromen schakelen zodat we met de arduino ook grotere motoren, lampen etc. kunnen aansturen.	Kijk goed op het schema welke contacten schakelen en welke van de spoel (de elektromagneet) zijn. Wij gebruiken het relais ook om de draairichting van een motor om te keren (de plus en de min verwisselen).	

Kleurcodes van weerstanden

Bij voorbeeld:
geel-groen-rood=4500 ohm (4,5k) (4k5)
oranje-oranje-bruin=330 ohm

Trouble Shooting

Wat doe je als je circuit niet werkt?
Prototyping met een breadboard is fantastisch om mee te experimenteren. Het is een hele snelle manier van werken. Nadeel is dat er al gauw een draden spaghetti ontstaat waardoor de kans op fouten toeneemt. In het begin worden verreweg de meeste vergissingen gemaakt bij het in het goede gaatje prikken van de draad. Als je rustig gaat controleren dan vind je vaak de fout wel maar het kan best zijn dat je keer op keer over dezelfde fout heenkijkt. Soms is het beter het circuit helemaal opnieuw op te bouwen. Controleer ook of je wel de goede onderdelen hebt, ze lijken vaak heel erg op elkaar.
Dan is het natuurlijk ook mogelijk dat je een kapot onderdeel of een kapotte draad hebt. Verwissel de onderdelen voor anderen als je vermoed dat er een kapot is.

Programmeren met Arduino

Om een arduino te programmeren heb je software nodig: de programmeeromgeving (afgekort IDE; integrated development environment). De arduino software is geschreven in java en gebaseerd op C/C++, processing, avr-gcc en andere open source hardware. Hieronder een kleine handleiding voor de arduino programmeertaal.

Structuur

Mensen communiceren onderling via taal. Als wij met een computer communiceren gebeurd dat ook door middel van een taal, programmeertaal, met heel eigen regels en grammatica. Van die regels mag je niet teveel afwijken, want dan begrijpt de computer je niet meer. Een arduino programma (vaak een scetch, schets, genoemd) bestaat uit minstens twee programma blokken: De void setup en de void loop.

void setup () { …… }

Aan het begin zie het vreemde woord 'void'. Het betekend zoiets als 'leegte'. Beschouw het maar *plaats* voor de setup. In de setup staat wat we allemaal gebruiken (welke variabelen, pinnen etc.). Net als het recept voor het bakken van een cake: We hebben nodig meel, eieren, kom, oven, etc..
Na het woord setup staan twee haakjes: die staan er nu eenmaal. Daarna komen twee gekrulde haakjes {accolades}: Hiertussen staat de informatie van de setup. Dit wordt bij het opstarten van het programma één keer gelezen en onthouden door de arduino.

void loop () { …… }

In de loop functie wordt beschreven wat er gedaan moet worden. Om bij de vergelijking met de cake te blijven: Doe het meel en eieren in de kom, goed mixen, 60 min in de oven, etc.. De instructies die in de void loop staan worden eindeloos herhaald tot de stroom wordt uitgezet. De loop begint en eindigt altijd met een accolade, maar binnen de loop kunnen kleinere blokjes programma staan die weer hun eigen paar accolades hebben. Bij het intypen van een programma kan je zo'n accolade nog wel eens vergeten. Als je in de ardiuno sketch naast een accolade klikt laat het programma je zien waar de tweede staat.

Speciale tekens

Arduino heeft een aantal tekens om coderegels, commentaren en codeblokken aan te geven.

; Puntkomma
Iedere instructie (programma regel) wordt afgesloten met een puntkomma. Vergeten puntkomma's zijn vaak oorzaak van niet compileren en frustratie.

{} Accolades. Worden gebruikt om aan te geven wanneer een blok met codes begint en eindigt. Binnen een blok kunnen meerdere kleine code blokken liggen met elk hun eigen accolades.

//Regel commentaar. Het programma negeert alles op die regel wat na de // komt. Het wordt gebruikt om commentaar te schrijven en ook om tijdelijk programma regels uit te schakelen.

/*….*/ Commentaar Blok
Wordt gebruikt voor commentaar over meerdere regels. Meestal gebruikt voor uitleg over het programma voor jezelf en anderen. Doen!

Variabelen

Variabelen kan je gebruiken om getallen op te slaan in het geheugen om later weer te kunnen gebruiken. Er zijn verschillende type's variabelen die meer of minder geheugenplek innemen, afhankelijk van het aantal getallen dat ze moeten opslaan.

int (integer): Het meest gebruikte data type. Heeft geen decimalen en kan een getal opslaan tussen -32.768 en 32.767

boolean Kan een van de twee waardes hebben: true (waar) of false (niet waar). Boolean gebruikt heel weinig geheugenruimte.

long Wordt gebruikt als integer niet groot genoeg is. Long kan getallen opslaan tussen –2.147.483.648 en 2.147.483.647.

float Komt van floating point: decimalen. Wordt gebruikt voor berekeningen met decimalen . Kost veel geheugenruimte.

char (character): Slaat een toetsenbordteken op in de ASCII code (bv A=65).

Constanten

Een aantal woorden zijn bij de arduino gekoppeld aan speciale, constante waardes. Let op de hoofdletters en kleine letters.

HIGH en **LOW** worden gebruikt om een arduino pin aan of uit te zetten.
Bij voorbeeld: digitalWrite(13, HIGH); // dit betekend: zet 5 volt op pin 13

input en **output** bepalen of een pin een input (spanning meten) of output (spanning geven) is.
Bij voorbeeld: pinMode(13, OUTPUT); // pin 13 is een output.

true en **false** controleren of iets waar of niet waar is. False (niet waar) is 0. Alle waardes die niet nul zijn beschouwt de arduino als true (waar).
Bij voorbeeld: if (b==true);
{ doe iets; }

Besturingsstructuren

if (conditie) {….}

Bij deze opdracht wordt gecontroleerd of de conditie die tussen de haakjes staat waar is. Als dat zo is wordt de actie uitgevoerd die tussen de accolades staat. Zoniet, dan wordt de actie overgeslagen.

```
Voorbeeld:  if (val == 1) {                  // als de waarde val 1 is
             digitalWrite(LED,HIGH);         // gaat de led aan
            }
```

if….else

Hier wordt gekozen tussen twee condities. Als de ene waar is, doe {…..} anders doe {——}

```
Voorbeeld:  if (inputPin == HIGH) {
            voer actie A uit;
            }
            else {
            voer actie B uit;
            }
```

Je kan het nog verder uitbreiden. Let goed op de haakjes en puntkomma's.

```
Voorbeeld:  if (inputPin < 500) {
            Voer actie A uit;
            }
            else if  (inputPin <= 1000)  {
            voer actie B uit;
            }
            else {
            voer actie C uit;
            }
```

for (variabele; voorwaarde; vermeerdering of vermindering;)

Laat je een stukje programma een bepaald aantal keren uitvoeren.

```
Voorbeeld:  for (int i = 0; i < 10; i ++)     // De variabele heet i en begint bij 0. De voorwaarde is dat
            {                                 // i kleiner is dan 10. i wordt telkens met 1 opgehoogd.
            Serial.print("hoi");
            }
```

while (variabele)

Lijkt wel wat op de for loop. Doe iets zolang aan een bepaalde voorwaarde voldaan wordt. Wordt er niet aan de voorwaarde voldaan dan wordt het blokje overgeslagen.

```
Voorbeeld:  var = 0;
            while (var < 200) {      // doe iets 200 keer
            var ++;
            }
```

Rekenen

+	Optellen
-	Aftrekken
*	Vermenigvuldigen
/	Delen
%	Modulo. Geeft het restgetal van een deling: 12 % 10 geeft 2

Logische Berekeningen

Logische berekeningen (Boolean operators) zijn vergelijkingen waarbij de uitkomst waar of niet waar is. Ze worden vaak gebruikt in **if** opdrachten om verschillende condities met elkaar te vergelijken:

&&	Logische en (and): Waar als beide vergelijkingen waar zijn.	if (x>0 && x<5)
||	Logische of (or): Waar als een van de twee waar is.	if (x>0 || y>0)
!	Logische niet (not): Waar als de vergelijking niet waar is.	if (!x > 0)

Analoge Input en Output

analogRead(pin)
Leest het voltage (tussen 0 en 5 volt) dat staat op een analoge input pin (pin 0 t/m 5 zijn analoge pinnen) en geeft daarvoor een waarde tussen 0 en 1023.
Voorbeeld:

```
val = analogRead(2); // Meet de spanning op pin2
                     // en geef een waarde (val)
```

analogWrite(pin, value)
Een Arduino is een digitaal apparaat en kan alleen wel of geen stroom zetten op een pin. De pinnen 3, 5, 6, 9, 10 en 11 ondersteunen PWM (pulse width modulation). Deze pinnen kunnen heel snel aan en uit gezet worden waardoor ze analoog lijken. De *value* kan een getal zijn tussen 0 en 255 omgeschaald wordt naar een spanning tussen 0 en 5 V.
Voorbeeld:

```
analogWrite(9, 128);  // Dim een led op pin 9 met
                      // 50%
```

In het arduino programma staan onder *file-examples* veel eenvoudige programmavoorbeelden.

Op de arduino site vind je onder *Reference* een uitgebreid overzicht van de programma taal.
http://arduino.cc/en/Reference/HomePage

Vergelijken

```
x==y    // x is gelijk aan y
x!=y    // x is niet gelijk aan y
x<y     // x is kleiner dan y
x>y     // x is groter dan y
x<=y    // x is kleiner of gelijk aan y
```

Digitale Input en output

pinMode(pin, mode)
Stelt een bepaalde pin in als output of als input.
Voorbeeld:

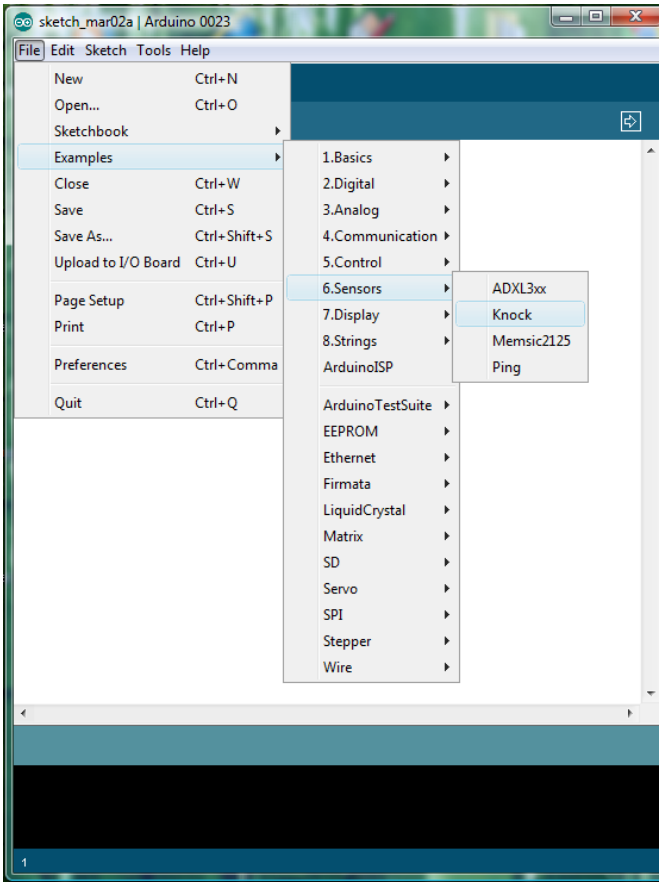
```
pinMode(7, INPUT); // Maakt van pin 7 een input.
```

digitalWrite(pin, value)
Zet een bepaalde pin hoog (HIGH) of laag (LOW).
Voorbeeld:

```
digitalWrite(9, HIGH); // Zet pin 9 aan.
```

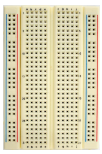
digitalRead(pin)
Leest of een pin hoog of laag staat.
Voorbeeld:

```
val = digitalRead(8) // Geeft de waarde HIGH (wel
                     // spanning) of LOW (geen
                     // spanning) van pin 8.
```



Les 0
Breadboard

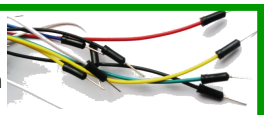
Wat heb je nodig?


Breadboard

Batterij 4,5v


Weerstand 330 ohm oranje/oranje/bruin


Testsnoeren

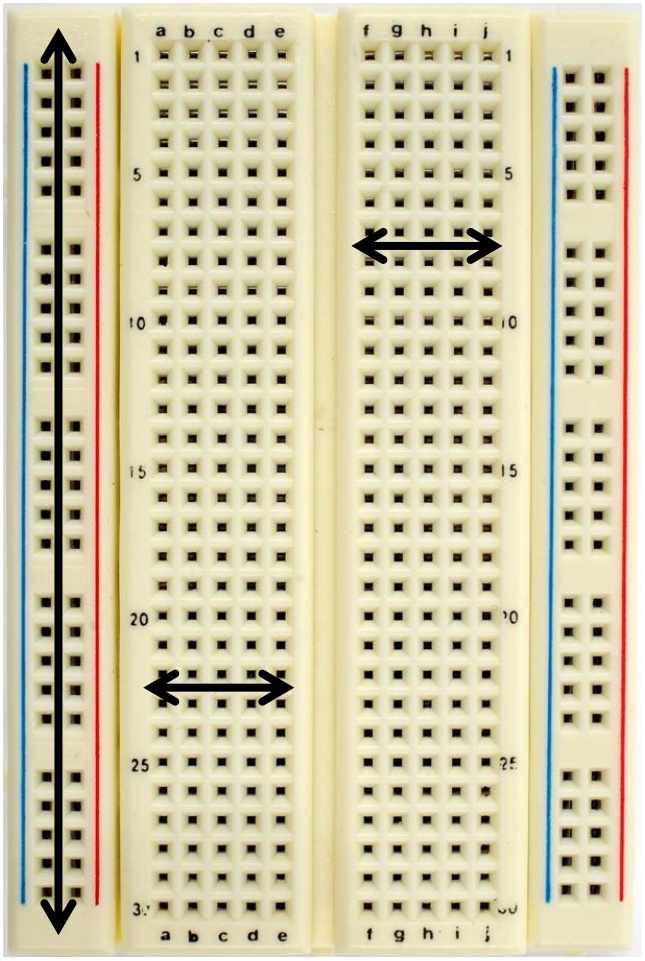

Jumper wire

Led

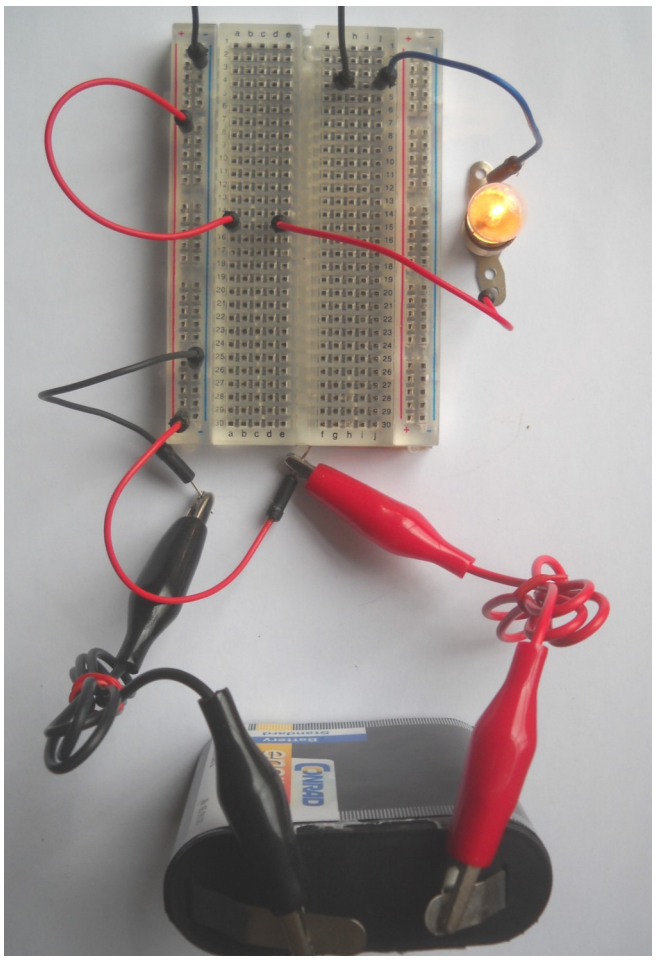
Wat gaan we doen?

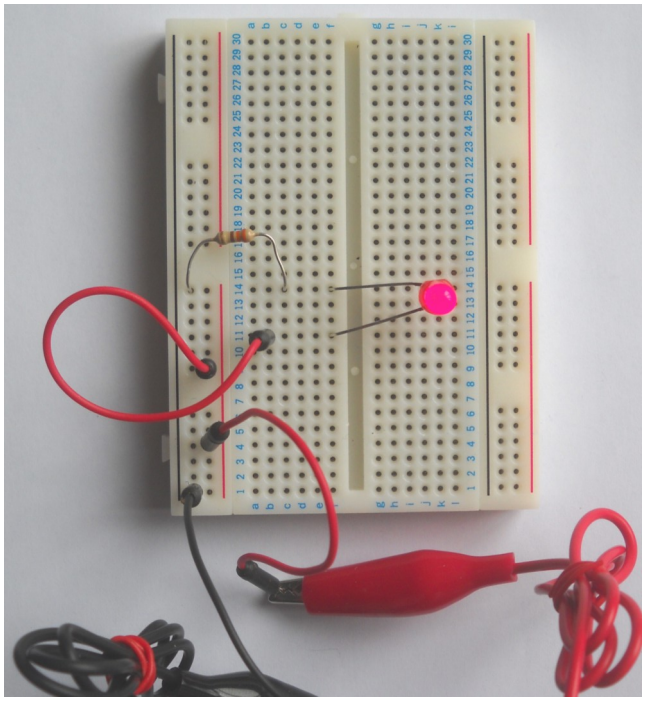
Als eerste gaan we het breadboard uitproberen. Ga met de batterij, draadjes en lampje onderzoeken welke gaatjes met elkaar zijn doorverbonden. Prik de jumpers in de verschillende gaatjes en kijk of je een stroomkring kunt maken en het lampje gaat branden. Verbind de plus (+) van de batterij met de gaatjes bij de rode lijn en de min (-) met de gaatjes bij de blauwe lijn. Gebruik zoveel mogelijk voor de plus rode draden en voor de min zwarte of blauwe.

Ga nu ook een led lampje aansluiten. Door een led kan de stroom maar in één richting, de plus en min moeten goed worden aangesloten: Het lange pootje aan de plus en het korte aan de min kant. Bij een led moet ook altijd een weerstand worden aangesloten om de spanning te verminderen (maximaal 3 volt). Verbind daarom de min kant van de led met een 330 ohm weerstand en verbind de weerstand met de min op het breadboard.



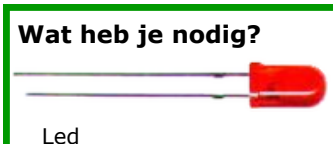
Bij het **breadboard** zijn gaatjes zijn doorverbonden in de richting van de pijlen



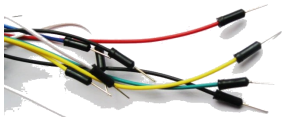


Les 1 a
Blink

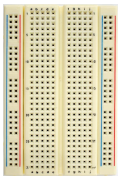
Wat heb je nodig?



Led



Jumper wire



Breadboard

Arduino

Weerstand 330 ohm oranje/oranje/bruin

Wat gaan we doen?

We beginnen met iets heel eenvoudigs: Een ledje laten knipperen. Zoek de onderdelen die boven staan bij elkaar en monteer ze zoals op de afbeelding rechts. Je doet eigenlijk hetzelfde als bij les 0 alleen nu met een arduino i.p.v. een batterij. De plus is de 5V pin en de min is de Gnd (ground). De pluskant van de led wordt verbonden met pin 13. Nu kan je een programma gaan uploaden naar de arduino.

Sluit de usb kabel aan en start Arduino op. Als de led meteen begint te knipperen stond er nog een programma op dat pin 13 gebruikt. Een programma blijft altijd op de arduino staan tot je er een andere opzet, ook als de stroom eraf is. Ga naar **File>Examples>1.Basics>Blink**

Als het goed is staat het programma (de sketch of ook vaak code genoemd) op je beeldscherm zoals de afbeelding rechtsonder. Dit programma laat een led lampje met een frequentie van 1 seconde aan en uit knipperen. Voor je een sketch gaat uploaden moet je eerst 'compileren' door op de *verify* knop te drukken: De arduino software gaat controleren of er 'grammaticale' fouten in de sketch zitten. Als alles in orde is krijg je beneden de melding 'done compiling' en dan kan je op de Upload knop drukken. Als beneden de melding 'done uploading' komt is de sketch in de arduino.

Verify

Upload

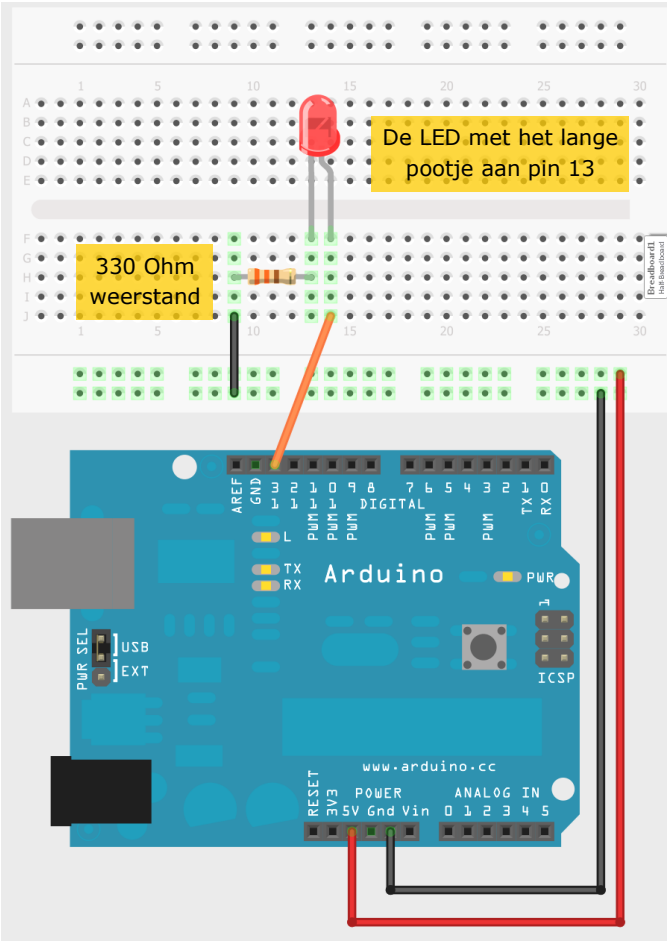
Wat betekenen nu al die cryptische mededelingen? Alles wat achter de dubbele slash // of tussen /* en */ staat is commentaar voor jezelf of anderen. De sketch begint met de 'void setup'. Hierin staat welke pin (nr. 13 hier) gebruikt wordt en dat het een 'output' is (een uitgang, er moet stroom uit komen): De Pinmode.

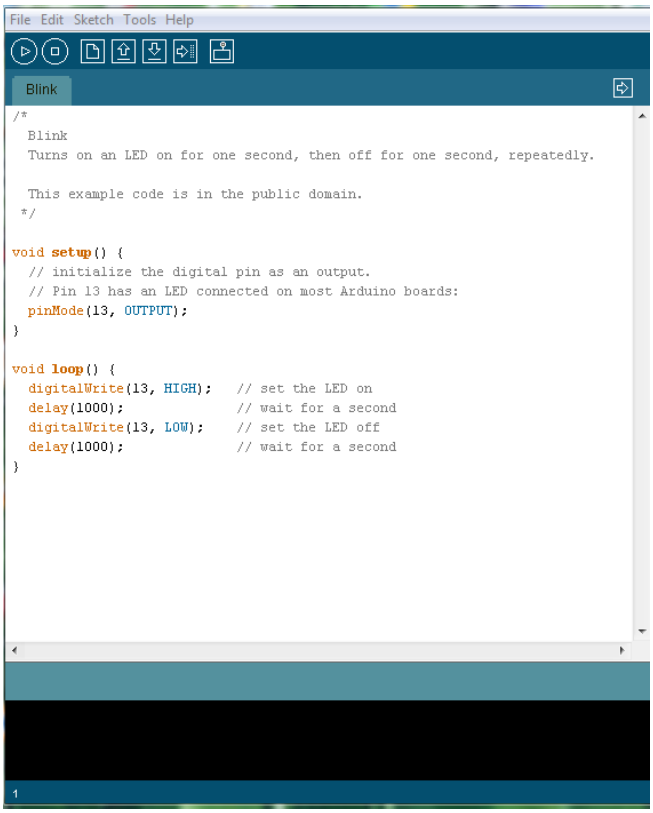
Daarna komt de 'void loop', het tweede gedeelte van de sketch dat steeds herhaald wordt. Eerst staat er: *digitalWrite(13,High).* Dat betekent: Zet spanning op pin 13. Dan komt *delay(1000):* Wacht 1000 milliseconde (1 sec.), de led brand 1 sec.. Daarna *digitalWrite(13,LOW):* De stroom gaat nu van pin 13 af en de led gaat uit. Ook deze situatie blijft 1 seconde *(delay(1000).*

Opdracht:

Verander de aan -en uit tijden in de sketch en bekijk het resultaat. Bewaar de veranderde code.

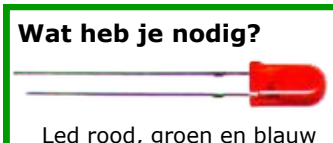
Meer informatie bij: Programmeren met arduino



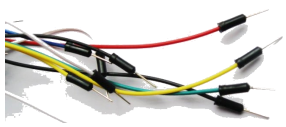


Les 1 b
Blink Blink Blink

Wat heb je nodig?

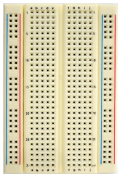


Led rood, groen en blauw



Jumper wire

Weerstand 330 ohm oranje/oranje/bruin 3x



Breadboard

Arduino

Wat gaan we doen?

Als één lampje leuk is zijn drie lampjes met verschillende kleuren natuurlijk nog leuker. Monteer de onderdelen zoals op de afbeelding hiernaast: Rode led op pin 11, groene op 10, blauw op 9, iedere led met een 330 ohm weerstand aan het korte pootje naar de min (Rnd).

Ga nu naar:

Sketches interactief ontwerpen

En open de sketch:

Les 1 rood geel blauw.

Verify en upload de sketch naar de Arduino.
Dit programma laat de rode en de groene led tegelijkertijd knipperen, een halve seconde aan, een halve seconde uit. Boven in de sketch zie je staan 'int rodePin = 11'. Zo stond het niet in de 'blink' sketch. Het is een andere manier die ook kan. De afkorting '**int**' betekend 'integer' en dat betekend dat het getal dat erachter komt een geheel getal is (zonder decimalen). Door de mededeling 'int rodePin=11' weet de arduino nu dat overal waar 'rodePin' geschreven staat, we eigenlijk pin 11 bedoelen. Dit maakt de sketch voor ons leesbaarder.

Opdracht:

Pas de code in de sketch aan zodat ook de blauwe led meeknippert. Tips nodig?
1–Wijs het pinnummer aan voor de blauwe led.
2–Zeg de arduino dat de blauwe led een output is.
3-Voeg twee regels toe in de loop.
Verify en upload je sketch. De drie led's knipperen nu tegelijkertijd aan en uit.

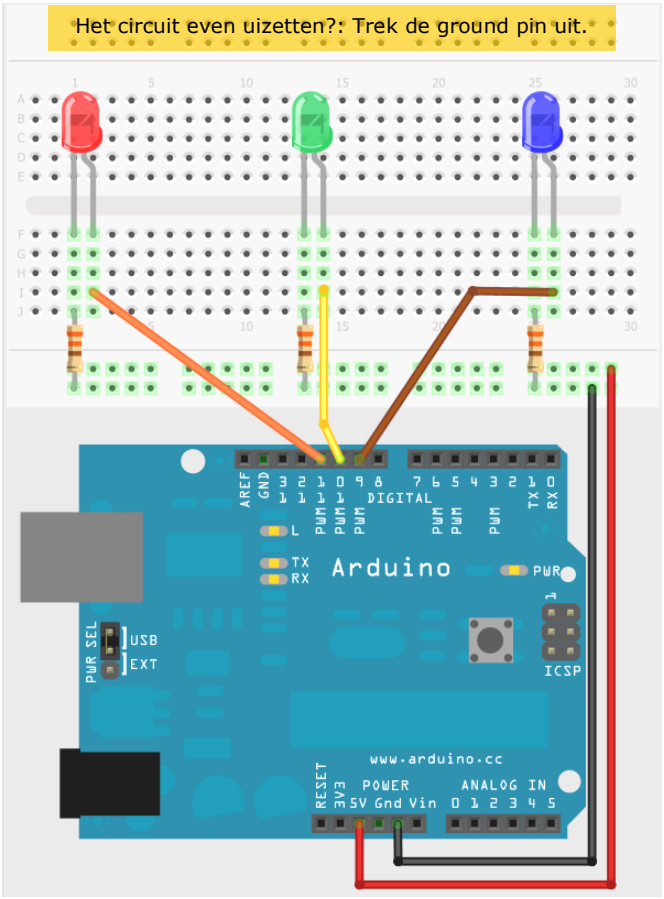
Opdracht:

1- Pas nu de 'loop' van de sketch zodanig aan dat de leds na elkaar knipperen. Tip: Om code regels van plek te veranderen of tussen te voegen kan je natuurlijk knippen/plakken gebruiken (Ctrl C/ Ctrl V). Je kan ook code regels 'uitschakelen' door er een dubbele slash voor te zetten.

2- Laat nu de led's ook sneller knipperen.

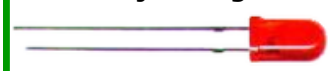
Bewaar al je aangepaste en gemaakte sketches.

Het kan zijn dat bij het 'compilen' de balk beneden bruin wordt en er een fout gevonden is. Vaak heb je dan een haakje of puntkomma of zo vergeten. Misschien heb je alleen maar pinmode in plaats van pinMode geschreven. Voor ons is dat niet zo'n verschil, maar voor een arduino zeker wel.

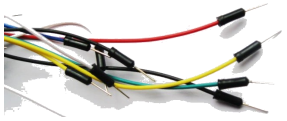


Les 1 c
Meer Kleur
Fade met For

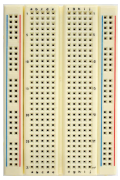
Wat heb je nodig?



Led rood, groen en blauw



Jumper wire



Breadboard

Arduino

Weerstand 330 ohm oranje/oranje/bruin

Wat gaan we doen?

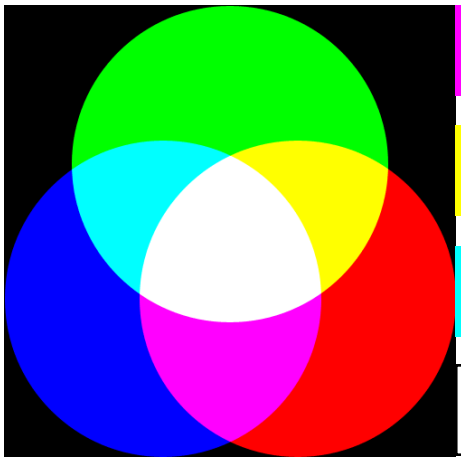
Nu we toch een rode een groene en een blauwe led hebben kunnen we nog wat meer met kleur gaan spelen. We kunnen met licht kleuren mengen. In het schema zie je dat we met de primaire kleuren rood groen en blauw de secundaire kleuren magenta (+/- violet) geel en cyaan (+/- turkoois) kunnen maken.

Opdracht:

Verander de code uit de vorige opdracht zo dat je achtereenvolgens rood, geel, groen, cyaan, blauw, en magenta maakt. Tussen iedere kleur moet een halve seconde pauze zijn.

Om het effect beter te kunnen zien kan je van een A4tje een doosje maken met een papieren zakdoekje of watjes erin om het licht diffuus te maken. Je kan ook fellere led's gebruiken.

Verander de naam en bewaar de sketch.

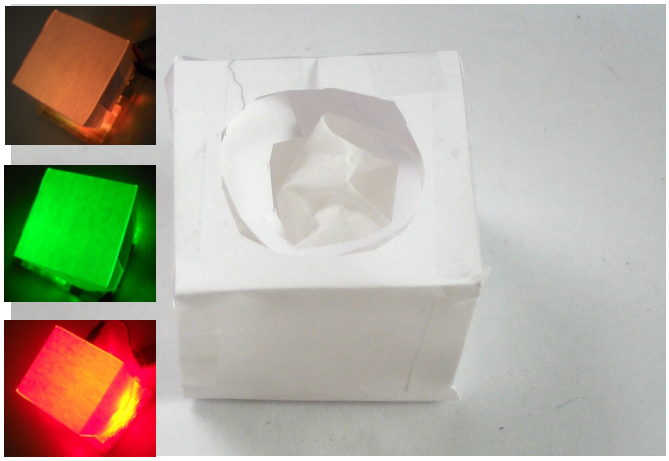


Rood + Blauw = Magenta

Rood + Groen = Geel

Groen + Blauw = Cyaan

Groen + Blauw + Rood = Wit



Wat kunnen we nog doen?

Nu we toch onze ledjes op pin 9, 10 en 11 hebben aangesloten kunnen we ook gaan dimmen. Bij deze pinnen en bij pin 3, 5 en 6 staat: ~ . Deze pinnen zijn uitgerust met Puls Width Modulation (PWM). De spanning op deze pinnen kan heel snel aan en uit gezet worden. Door de aan puls langer ('wijder') of korter te maken staat er langere of kortere tijd spanning (altijd 5 V) op de pin. Dit heeft het hetzelfde effect als meer of minder spanning zetten op de pin. Zo kunnen we een lampje harder of zachter laten branden of een motortje sneller of langzamer laten draaien.

Ga nu naar:

Sketches interactief ontwerpen

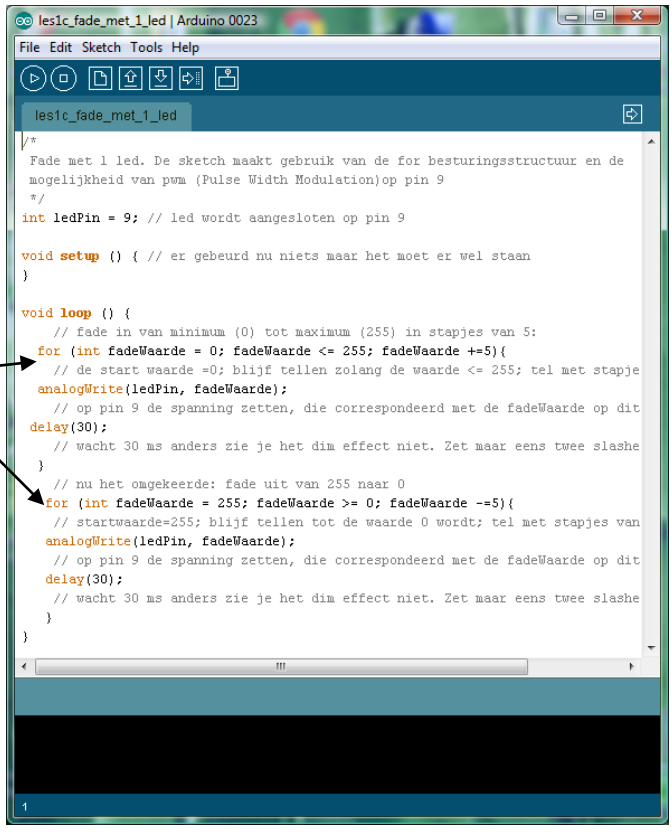
En open de sketch:

Les 1c fade met 1 led

We maken gebruik van de **for** loop. Die bestaat uit drie gedeeltes: for(startwaarde; controle waarde; stapgrootte). We beginnen met 0 (0 volt, lampje uit), dan gaan we telkens met 5 omhoog (+=5) tot het getal 255 is (5V lampje brand maximaal). Bij analogWrite(ledPin, fadeWaarde) krijgt pin 9 de hoeveelheid spanning (pulsen) die correspondeert met het getal van de fadeWaarde.

Opdracht:

Onderzoek welke veranderingen ontstaan door het veranderen van de stapgrootte en de delay. Drie ledjes faden? Ga naar de Arduino sketchmap en open *les1c fade samen*.



les1c_fade_met_1_led | Arduino 0023

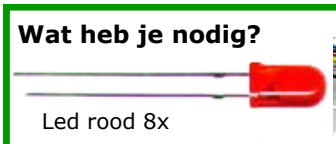
File Edit Sketch Tools Help

les1c_fade_met_1_led

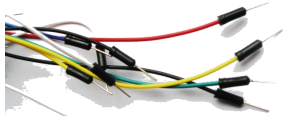
```
/*
 Fade met 1 led. De sketch maakt gebruik van de for besturingsstructuur en de
 mogelijkheid van pwm (Pulse Width Modulation)op pin 9
 */
int ledPin = 9; // led wordt aangesloten op pin 9

void setup () { // er gebeurd nu niets maar het moet er wel staan
}

void loop () {
    // fade in van minimum (0) tot maximum (255) in stapjes van 5:
  for (int fadeWaarde = 0; fadeWaarde <= 255; fadeWaarde +=5){
    // de start waarde =0; blijf tellen zolang de waarde <= 255; tel met stapje
   analogWrite(ledPin, fadeWaarde);
    // op pin 9 de spanning zetten, die correspondeerd met de fadeWaarde op dit
  delay(30);
    // wacht 30 ms anders zie je het dim effect niet. Zet maar eens twee slashe
  }
    // nu het omgekeerde: fade uit van 255 naar 0
  for (int fadeWaarde = 255; fadeWaarde >= 0; fadeWaarde -=5){
    // startwaarde=255; blijf tellen tot de waarde 0 wordt; tel met stapjes van
   analogWrite(ledPin, fadeWaarde);
    // op pin 9 de spanning zetten, die correspondeerd met de fadeWaarde op dit
   delay(30);
    // wacht 30 ms anders zie je het dim effect niet. Zet maar eens twee slashe
   }
}
```

Les 1 d
For nog een keer

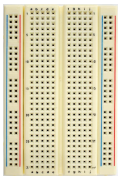
Wat heb je nodig?



Led rood 8x



Jumper wire



Breadboard

Arduino

Weerstand 330 ohm oranje/oranje/bruin 8x

Wat gaan we doen?

Nog meer led's. In een gewone taal kan je de zelfde gebeurtenis op veel manieren beschrijven. In computertaal is het net zo. We willen nu 6 ledjes achtereenvolgens laten branden. We kunnen dan een sketch beginnen met alle led's aan pinnen toe te wijzen (int ledPin1=2 etc. etc.) vervolgens in de setup alle pinnen output maken (pinmode ledPin1, OUTPUT etc etc) en tenslotte ze in de loop allemaal HIGH en LOW laten worden (digitalWrite ledPin1, HIGH etc. etc.). Al met al wordt het een omvangrijk programma. Dat kan ook anders, korter: Met onze *for* loop van de vorige pagina.
Ga nu naar:

Sketches interactief ontwerpen

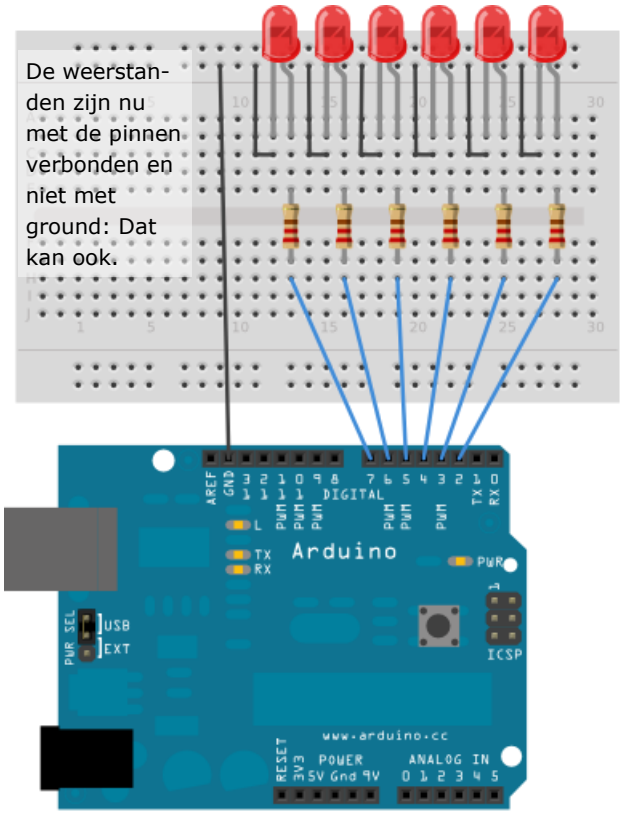
En open de sketch:

Les 1d looplicht

Bekijk de code eens en probeer te zien wat zal gebeuren. De *for* loop wordt gebruikt om te tellen. In de vorige sketch telde we getallen van 0 tot 255 die werden omgezet naar spanningspulsen. Nu worden de pinnen geteld. De ledjes zijn aangesloten op de pinnen 2 t/m 7. Die worden achtereenvolgens geteld en worden high en low gezet.
Bekijk de instructie :
for (int thisPin = 2; thisPin < 8; thisPin++)
De pin wordt hier een variabele die '*thisPin*' heet. Als het programma de eerste keer de loop doorloopt is de waarde van *thisPin 2* (pin 2 dus). Dan wordt er gecontroleerd of *'thisPin'* kleiner is dan 8. Wordt *'thisPin'* 8, dan stopt deze loop en gaat het programma verder. Tenslotte wordt de stapgrootte aangegeven en die is hier '++' en dat betekent dat er telkens met 1 stap omhooggegaan wordt (dus eerst pin 2, dan 3,4,5,6,7). De volgende *for* loop werkt precies andersom: We beginnen bij pin 7, steeds 1 naar beneden ('--'), tot en met pin 2 (zodat de ledjes in de omgekeerde volgorde aan en uit gaan).
Er zit nóg een slimmigheidje in de sketch. Helemaal boven staat: *int timer = 100*. Hiermee wordt de timer variabel gemaakt en hoef je alleen het getal te veranderen en overal waar *delay(timer)* staat wordt nu automatisch het nieuwe getal gelezen (probeer uit!). Monteer nu de ledjes zoals op de afbeelding rechtsboven en upload de sketch.

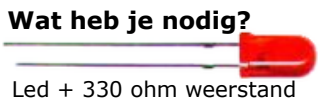
Opdracht:

Zet nog twee lampjes erbij en pas de sketch aan.

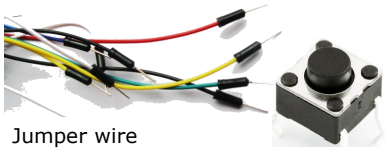


Les 2
Knop
if...else

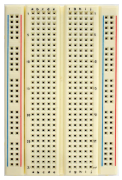
Wat heb je nodig?


Led + 330 ohm weerstand


Weerstand 10 Kohm bruin/zwart/oranje 2x


Jumper wire

Drukknop 2x


Breadboard


Arduino

Wat gaan we doen?

We gaan een knopje indrukken en daarmee een lampje laten branden. Niets bijzonders zou je zeggen, maar dat is het wel. Hierdoor maken we kennis met een van de belangrijkste besturingsstructuren: Het if commando (statement in het Engels). We kunnen zeggen: Als dit het geval is (de knop is ingedrukt bij voorbeeld), doe dan dat (laat de led branden bij voorbeeld). Maar je zou ook kunnen zeggen: Als de knop *niet* is ingedrukt en ook kunnen de knop en de led natuurlijk heel andere dingen zijn.

Monteer nu de onderdelen zoals op de afbeelding hiernaast en upload het onder geprinte programma: **les2 knop**.

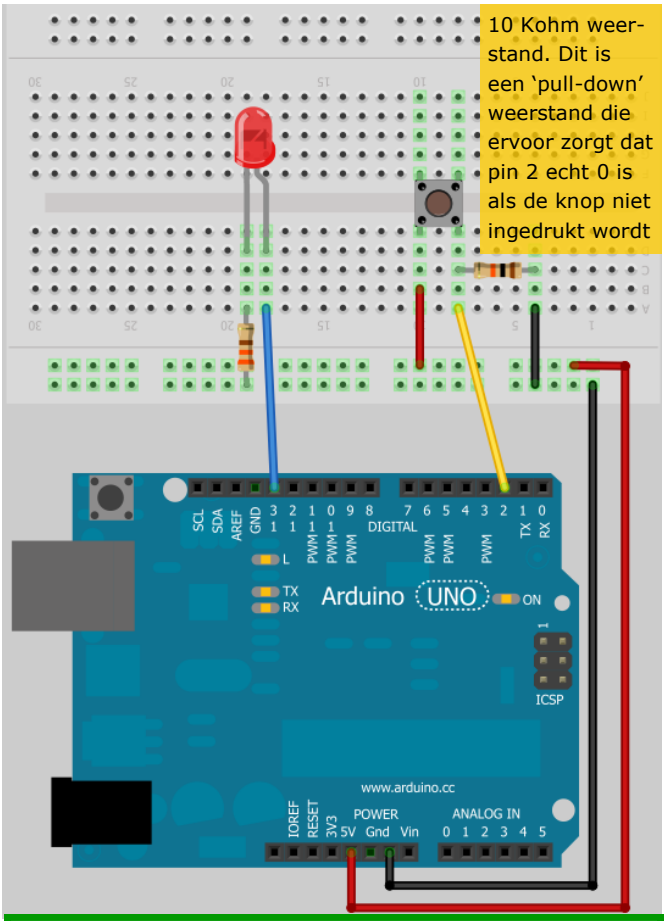
De knop is verbonden met pin 2, de led met pin 13. Er wordt ook een variabele vastgesteld voor de toestand van de knop '*int toestandKnop = 0'* (die kan 'hoog' of 'laag' zijn: 1 of 0). De beginwaarde is 0. In de loop wordt eerst de toestand van de knop gelezen (is pin 2 hoog of laag?): *'digitalRead (knopPin)'.* Dan komt de if structuur: Als *(if)* de waarde *toestandKnop* hoog is, dan moet de led aan. Anders *(else)* is de led uit.

Opdracht:

Verander de sketch zo dat de led uit gaat als je de knop indrukt. Tip: Dit kan op twee manieren. Een manier is om de vergelijking na *if* te veranderen (kijk op pagina 18 bij vergelijken). De tweede manier kan je zeker zelf vinden. Probeer ze beiden.

```
/* Knop
Als op de knop (verbonden met pin2) gedrukt wordt gaat de led
(verbonden met pin 13)aan.
*/
 int knopPin = 2;       // knop aan pin 2
 int ledPin =  13;      // led aan pin 13
 int toestandKnop = 0;  // variabele voor het lezen van de knop

void setup() {
  pinMode(ledPin, OUTPUT);     //ledpin is output
  pinMode(knopPin, INPUT);     //knop is input
}
void loop(){
  toestandKnop = digitalRead(knopPin); //toestandKnop is de waarde van knopPin
  if (toestandKnop == HIGH) {          //controleer of de knop ingedrukt is
    digitalWrite(ledPin, HIGH);        //indien ingedrukt: led aan
  }
  else {
    digitalWrite(ledPin, LOW);         //anders: led uit
  }
}
```

We kunnen de arduino ook twee 'toestanden' in de gaten laten houden. Plaats op het breadboard nog een knop erbij en verbind die met pin 3. Upload **les2 twee knoppen.** Als je nu op de ene knop drukt gaat de led aan, met de andere gaat hij uit. We hebben nu twee if statements na elkaar. Kijk ook op pagina 7 bij besturingsstructuren.



10 Kohm weerstand. Dit is een 'pull-down' weerstand die ervoor zorgt dat pin 2 echt 0 is als de knop niet ingedrukt wordt

Druk op de serial monitor knop. Je kan nu op het beeldscherm de toestand van de knop zien.

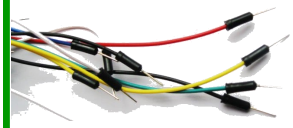
les2_twee_knoppen §

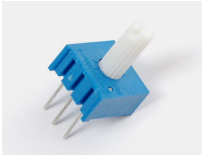
```
/* les2a twee knoppen.
Indien er geen knop ingedrukt wordt, dan blijft de led in de
stand van de knop die het laatst ingedrukt werd.
De twee regels die beginnen met 'Serial' zorgen ervoor dat de
serial monitor gestart wordt en de waarde geprint wordt op het
beeldscherm.
 */
 int knop2 = 2;                  // knop 2 verbonden met pin 2
 int knop1 = 3;                  // knop 1 verbonden met pin 3
 int ledPin =  13;               // led verbonden met pin 13

void setup() {
 Serial.begin(9600);    // start de serial monitor (laatste knop)

  pinMode(ledPin, OUTPUT);       // ledpin 13 is output
  pinMode(knop2, INPUT);         // pin 2 (knop 2) is input
  pinMode(knop1, INPUT);         // pin 3 (knop 1) is input
}

void loop(){
  int toestandKnop2 = digitalRead(2); // 'lees' de waarde op pin 2
  Serial.println(toestandKnop2);      // laat de waarde van pin 2
                                      // (0 of 1) zien op de monitor
  if (digitalRead(knop1) == HIGH) {   // als knop1 ingedrukt wordt
      digitalWrite(ledPin, LOW);      // led uit
  }
  if
      (digitalRead(knop2) == HIGH){   // als knop 2 ingedrukt wordt
      digitalWrite(ledPin, HIGH);     // led aan
  }
}
```

Les 3

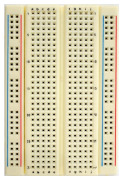
Servo Motor

Wat heb je nodig?


Jumper wire


Potmeter


Servo motor


Breadboard


Arduino

Wat gaan we doen?

Het wordt tijd om wat te gaan laten bewegen. Een servo motor is daarvoor vaak heel geschikt. De servo's die wij gebruiken zijn kleine elektromotoren met tandwielen die de snelheid verlagen maar de kracht vergroten. Bovendien kunnen ze elektronisch bestuurd worden. De standaard servo draait niet rond, maar kan maximaal 180 graden draaien. De positie wordt bepaald door de lengte van pulsen, tussen 1,25 milliseconden(0 graden) en 1,75 msec (180 graden), die de arduino naar de servo stuurt (1,5 msec is dus 90 graden). Voor de servo besturing is een aparte 'bibliotheek' gemaakt: software die het eenvoudiger maakt om de servo te programmeren. Daardoor kunnen we een getal tussen 0 en 180 invoeren om de servo tussen 0 en 180 graden te positioneren.

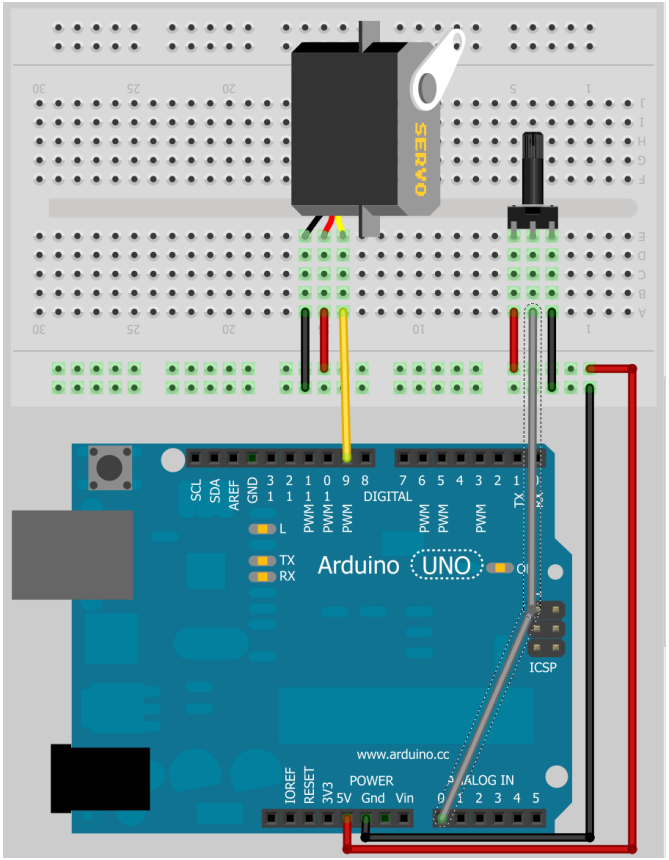
Monteer nu de servo op het breadboard zoals op de afbeelding hiernaast. De rode draad is de 5v aansluiting. De zwarte of bruine kabel (afhankelijk van de fabrikant) gaat naar Gnd (ground) en de kabel naar de pin is geel oranje of wit. Op het breadboard is ook nog een potmeter gemonteerd, die is voor de volgende sketch, maar je kan hem alvast monteren (het middelste pootje moet aan de analoge pin 2).

Upload nu de sketch **les3 sweep.**

De servo moet nu zo'n 180 graden heen en weer gaan.

Opdracht:

Verander de scetch zo dat de servo nog maar 45 graden op en neer gaat, maar wel ook sneller.



We kunnen de servo ook direct besturen met de potmeter (draaibare weerstand) die je al gemonteerd hebt. De potmeter is analoog en moet aan een analoge pin. Als je aan de potmeter draait geeft dat een waarde tussen 0 en 1023 op de pin. De servo heeft een getal nodig tussen 0 en 180. Dit kunnen we met de *map* instructie heel eenvoudig omschalen: map(val, 0, 1023, 0, 179).

Upload nu de sketch **les3 servo met potmeter** en probeer het uit.

les3_sweep

```
// Sweep: De servo motor gaat 180 graden heen en weer.

#include <Servo.h>// maak gebruik van de servo library (bibliotheek)
Servo mijnServo;  // we gebruiken een servo die we mijnServo noemen
int pos = 0;// de variabele voor de positie van de servo (start bij 0)

void setup()
{
  mijnServo.attach(9);  // de servo is verbonden met pin 9
}

 void loop()
{
  for(pos = 0; pos < 180; pos += 1)// ga van 0 tot 180 graden in stapjes
  {                                // van 1 graad
    mijnServo.write(pos);   // ga naar de positie van de variabele 'pos'
    delay(15);              // geef de servo 15 ms de tijd om te reageren
  }
  for(pos = 180; pos>=1; pos-=1)     // ga nu van 180 graden terug naar 0
  {
    mijnServo.write(pos);   // ga naar de positie van de variabele 'pos'
    delay(15);              // geef de servo 15 ms de tijd om te reageren
  }
}
```

les3_servo_met_potmeter

```
/*een servo cortroleren met een potmeter. De servo op pin 9, de potmeter
*/op de analoge pin 0.

#include <Servo.h>
Servo myservo;  // we gebruiken een servo die we myservo noemen
int potpin = 0; // de potmeter is verbonden met analoge pin 0
int val;        // val is de waarde van pin 0

void setup()
{
  myservo.attach(9);  // servo verbonden met pin 9
}

void loop()
{
  val = analogRead(potpin);// lees de waarde van pin 0 (= tussen 0 en 1023)
  val = map(val, 0, 1023, 0, 179);// schaal de waarde van pin 0 om naar een
                                  // waarde tussen 0 en 180
  myservo.write(val);             // ga naar de positie 'val'
  delay(15);                      // tijd voor de servo om te reageren
}
```

Les 3

Meer Servo Motoren

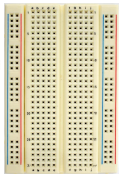
Wat heb je nodig?


Jumper wire


Drukknop 2x


Batterij 4,5v


Servo motor 2x


Breadboard


Arduino

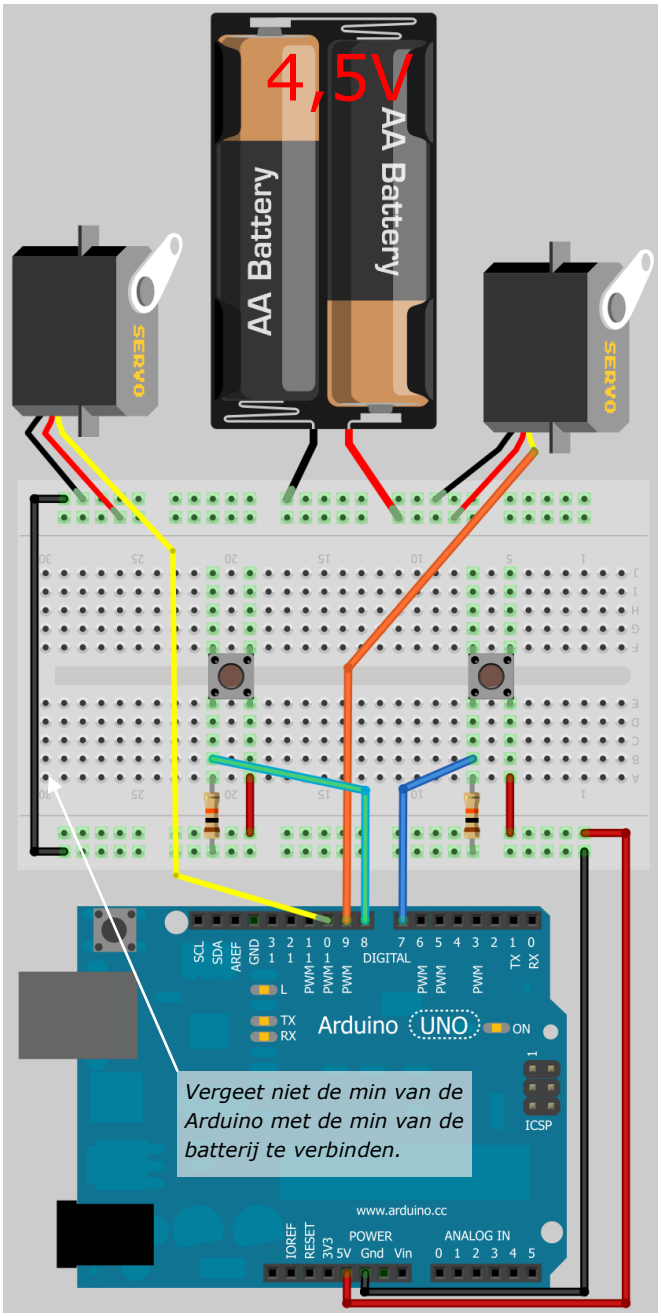

Weerstand 10 Kohm bruin/zwart/oranje 2x

Wat gaan we doen?

We kunnen ook meerdere servo's aansluiten (maximaal 12) maar dan kunnen we de stroom niet meer van de 5 volt aansluiting van de Arduino halen. Servo motoren verbruiken daarvoor teveel stroom. De plus en de min van de servo's moeten worden aangesloten op een aparte batterij. De draad (geel, oranje of wit) voor de stuurstroom wordt aangesloten op de pin van de Arduino. Wel moeten de min van de batterij en de min (Gnd) van de Arduino met elkaar verbonden zijn. We gaan de servo's nu bedienen met drukknoppen maar dit zouden natuurlijk ook andere sensoren kunnen zijn. De drukknoppen zijn verbonden met pin 7 en 8 en hebben een 'pull-down' weerstand. De servo's zijn verbonden met pin 9 en 10. Monteer de onderdelen zoals op de afbeelding rechts.

les3_twee_servo_s_met_knoppen

```
// Met deze sketch worden twee servo's gecontroleerd met
// twee knoppen.
 #include <Servo.h>
 #define knop1 7 // knop1 naar pin 7
 #define knop2 8 // knop2 naar pin 8
 Servo servo1;   // een servo heet servo1
 Servo servo2;   // een servo heet servo2
 int val1 = 0; //val1 is de waarde van knop1 (0 of 1)
 int val2 = 0; //val2 is de waarde van knop2 (0 of 1)
 int pos1 = 0; //de positie van servo1 (tussen 0 and 180)
 int pos2 = 0; //de positie van servo1 (tussen 0 and 180)
               //0 is steeds de startwaarde
 void setup () {
  servo1.attach(9);  // servo1 verbonden met pin 9
  servo2.attach(10); // servo2 verbonden met pin 10
  pinMode (knop1, INPUT); // knop1 is een input
  pinMode (knop2, INPUT); // knop2 is een input
 }
 void loop () {
  val1 = digitalRead (knop1); // lees knop1
  if (val1==HIGH) {//val1 is high als pin 7 hoog is (1 is)
    pos1=170;         //dus als knop 1 is ingedrukt
    servo1.write(pos1); // ga naar positie 170 graden
    delay(15);}
  else {
    pos1=10;
    servo1.write(pos1); // ga naar positie 10 graden
    delay(15);}

  val2 = digitalRead (knop2); // lees knop2
  if (val2==HIGH) {//val2 is high als pin 8 hoog is (1 is)
    pos2=170;          //dus als knop 2 is ingedrukt
    servo2.write(pos2); // ga naar positie 170 graden
    delay(15);}
    else {
    pos2=10;
    servo2.write(pos2); // ga naar positie 10 graden
    delay(15);}
   }
```



Upload: **les3 twee servo's met knoppen.**
Kijk even naar de tweede sketchregel: *#difine knop1 7* (let op, zonder puntkomma). Dit is alleen maar een andere schrijfwijze van *int knop1 = 7*.
Probeer het programma uit. Als het niet goed werkt loop dan alle verbindingen nog eens na.
Met servo's kan je heel veel leuke dingen doen. Kijk eens rond op internet en youtube en doe inspiratie op. Opdracht suggesties: Maak een percussie muziekmachine met meerdere servo's. Of verplaats een voorwerp met 'pan/tilt' servo's met grijper.
Je kan ook doorgaan met het volgende onderdeel.

Les 4

FSR

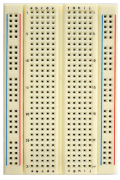
Wat heb je nodig?


Jumper wire


Led + 330 ohm weerstand


Weerstand 10 Kohm bruin/zwart/oranje


FSR


Breadboard

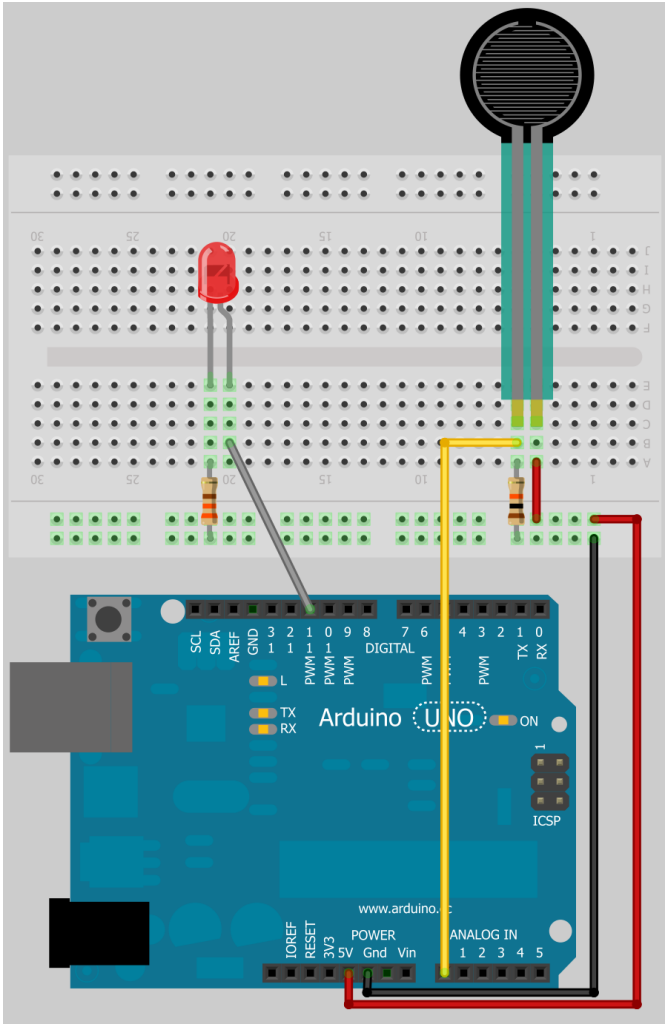

Arduino

Wat gaan we doen?

We gaan een sensor gebruiken om druk waar te nemen: Een FSR (Force Sensitive Resistor). Als je daarin knijpt wordt de weerstand minder en door de schakeling die we gebruiken, een 'spanningsdeler', meten we op pin 0 meer spanning. De spanning op een pin kan variëren tussen 0 en 5 volt en dit wordt door de Arduino omgezet in een getal tussen 0 (0 volt) en 1023 (5 volt). Kortom: Hoe meer druk, hoe hoger het getal. Kijk voor meer uitleg op: http://www.ladyada.net/learn/sensors/fsr.html.

Monteer de onderdelen zoals op de afbeelding hiernaast. Met deze schakeling en sketch (rechtsonder) kunnen we op twee manieren zien of de FSR werkt. Er is een led aangesloten op pin 11 (met pwm) en de helderheid is afhankelijk van de fsrwaarde (die wordt wel met *map* van 0 tot 1023 naar van 0 tot 255 omgeschaald). We gebruiken ook de serial monitor  om te zien wat precies de 'fsrWaarde' is.

Upload: **les4 fsr test sketch** (afb. rechts onder**),** start de serial monitor (De laatste knop in de menubalk) en druk op de sensor.



We kunnen zo ook met tekst aangeven hoeveel druk er is. Met de *if...else* constructie kunnen we zeggen: Als de fsrwaarde kleiner is dan … print dan … Je snapt al dat je op deze manier niet alleen iets kunnen printen op het beeldscherm, maar net zo goed bij voorbeeld (servo)motoren kan aansturen(pag. 18). Upload: **les4 simpele drukmeting.**

```
// FSR simpele drukmeting
int fsrPin = 0;//FSR en 10Kohm verbonden met analoog 0
int fsrWaarde; // meting van analoge pin 0
 void setup(void) {
  Serial.begin(9600);
}
 void loop(void) {
  fsrWaarde = analogRead(0);
 Serial.print("Analoge waarde = ");
 Serial.print(fsrWaarde);// druk de analoge waarde af
    if (fsrWaarde < 10) {
    Serial.println(" - Geen druk");
  } else if (fsrWaarde < 200) {
    Serial.println(" - Lichte aanraking");
  } else if (fsrWaarde < 500) {
    Serial.println(" - Lichte druk");
  } else if (fsrWaarde < 800) {
    Serial.println(" - Matige druk");
  } else {
    Serial.println(" - Grote druk");
  }
  delay(1000);
}
```

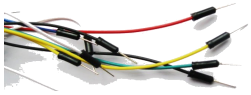
les4_fsr_test_sketch

```
/* FSR test sketch.
Een contact van de FSR is verbonden met 5V. Het andere contact
is verbonden met de analoge pin 0 en via een 10 Kohm weerstand
met de min (Gnd).Led verbonden met pin 11
Voor meer informatie www.ladyada.net/learn/sensors/fsr.html
*/

int fsrAnalogePin = 0; // FSR is verbonden met analoog 0
int LEDpin = 11;      // De LED is verbonden met pin 11 (pmw pin)
int fsrWaarde;       // De analoge waarde van de fsr spanningdeler
int LEDhelderheid;  // De helderheid van de led tussen 0 en 255

void setup() {
  Serial.begin(9600);   // start de serial monitor
  pinMode(LEDpin, OUTPUT);
}

void loop() {
  fsrWaarde = analogRead(fsrAnalogePin);
  Serial.print("Analoge waarde = "); // print 'Analoge waarde'
  Serial.println(fsrWaarde); // print de fsrwaarde op de monitor

  // maak van getallen tussen 0 en 1023 getallen tussen 0 en 255
  LEDhelderheid = map(fsrWaarde, 0, 1023, 0, 255);
  analogWrite(LEDpin, LEDhelderheid);

  delay(100);
}
```

Les 4

FSR met Servo

Wat heb je nodig?



Jumper wire

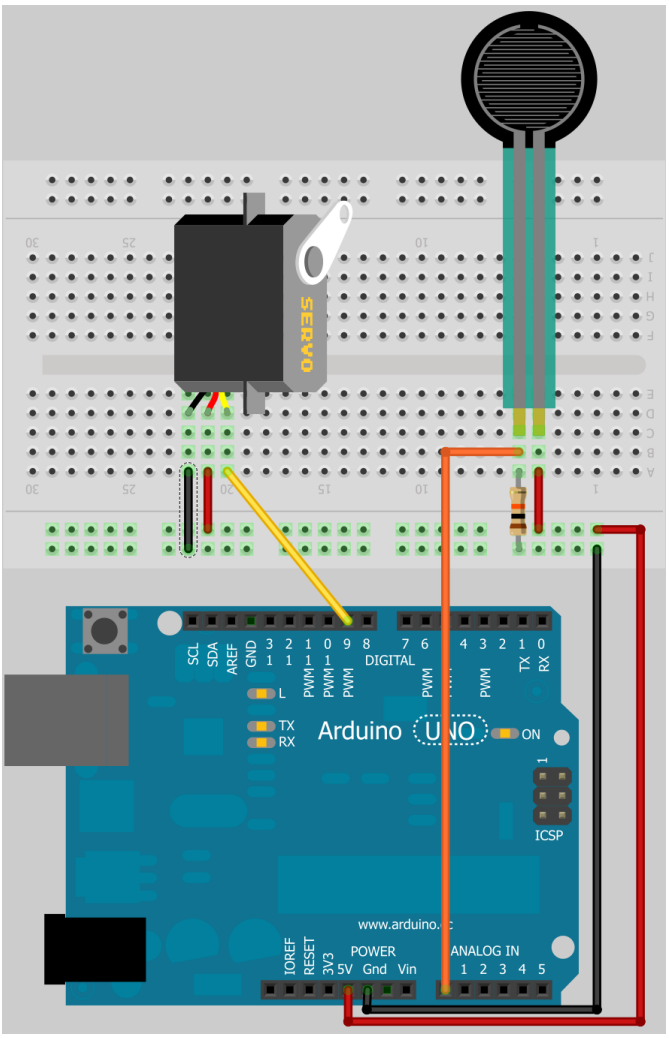
FSR

Weerstand 10 Kohm bruin/zwart/oranje

Servo motor

Wat gaan we doen?

We hebben nu al veel gedaan en een knipperend ledje is inmiddels kinderspel. Het wordt nog interessanter als we programma's gaan combineren, de mogelijkheden zijn dan eindeloos. We gaan een combinatie maken van de sketches die we gebruikt hebben voor de FSR en voor de servo. Daarbij gaan we ook weer gebruik maken van de serial monitor om zichtbaar te maken wat we doen en ook de *if* constructie zien we hier terug. De bedoeling is om een sorteerapparaat te maken waarbij de lichte gewichten in het ene bakje komen en de zwaardere in het andere. Als de druk heel weinig is, dan is de weger leeg en moet de servo in de midden stand staan (90 graden). Bij een lichte druk moet de servo naar een kant gaan (5 graden) en is er veel druk, dan gaat de servo de andere kant uit (175 graden) en valt het voorwerp in het andere bakje. Is het voorwerp te zwaar dan blijft de servo op 90 graden staan. Als je de onderstaande sketch bekijkt dan zie je dat de sketch *les4 simpele drukmeting* het uitgangspunt was. Aan iedere fsrWaarde is nu een stand (*pos*) van de servo gekoppeld.



les4_fsr_met_servo §

```
// FSR met servo. Sorteren op basis van gewicht.
#include <Servo.h>
Servo servo;   // de servo heet servo
int pos = 0;   // De positie van de servo (tussen 0 en 180).
int fsrPin = 0;//FSR en 10Kohm verbonden met analoog 0
int fsrWaarde; // meting van analoge pin 0
 void setup(void) {
   servo.attach(9);  // servo verbonden met pin 9
  Serial.begin(9600);// start de serial monitor
}
 void loop(void) {
  fsrWaarde = analogRead(0);
 Serial.print("Analoge waarde = ");
 Serial.print(fsrWaarde);// druk de analoge waarde af
    if (fsrWaarde < 10) {
    Serial.println(" - Weger leeg");
     pos=90;
     servo.write(pos); // ga naar positie 90 graden
  } else if (fsrWaarde < 200) {
    Serial.println(" - Licht voorwerp");
     pos=5;
     servo.write(pos); // ga naar positie 5 graden
  } else if (fsrWaarde < 500) {
    Serial.println(" - Zwaar voorwerp");
     pos=175;
     servo.write(pos); // ga naar positie 175 graden
  } else if (fsrWaarde >= 500) {
    Serial.println(" - Te zwaar voorwerp");
     pos=90;
     servo.write(pos); // ga naar positie 90 graden
  }
  delay(1000);
}
```

Opdracht: Monteer de onderdelen zoals op de afbeelding hierboven en upload: **les4 FSR met servo.** Maak ook met eenvoudige middelen een werkende sorteerder (zoek de juiste fsr waardes!).

Een van de vele manieren waarop een sorteerder gemaakt kan worden. Boekensteun, wijnglazen, lepeltje en kaarsbakje.

Les 5
DC Motor met Transistor

Wat heb je nodig?

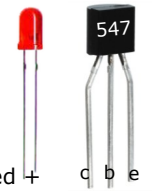

DC Motor

Batterij 4,5

Condensator 100µF


1 KΩ bruin/zwart/rood

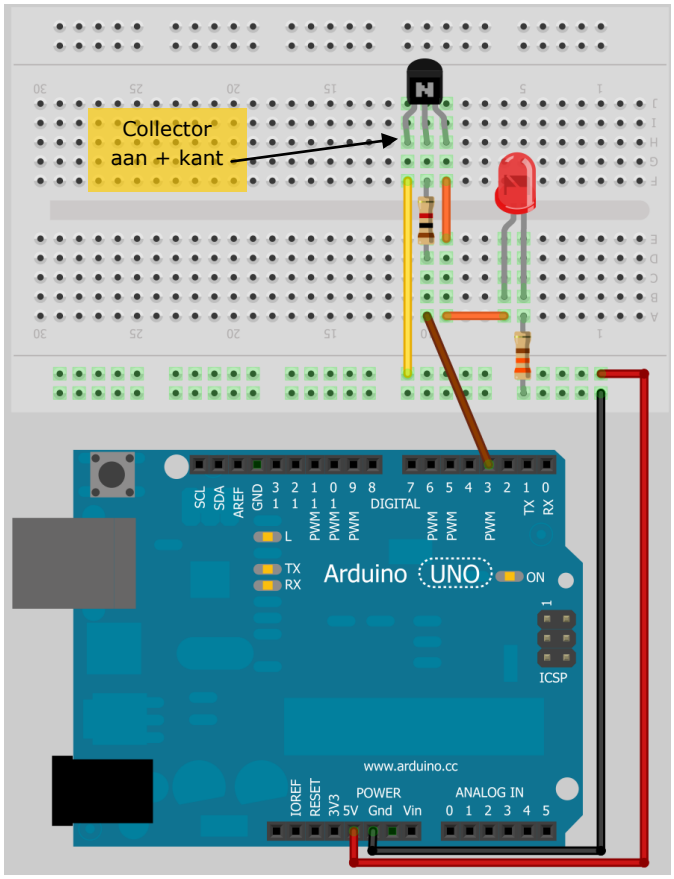

Diode 1N4001



Led + 330 Ω Transistor

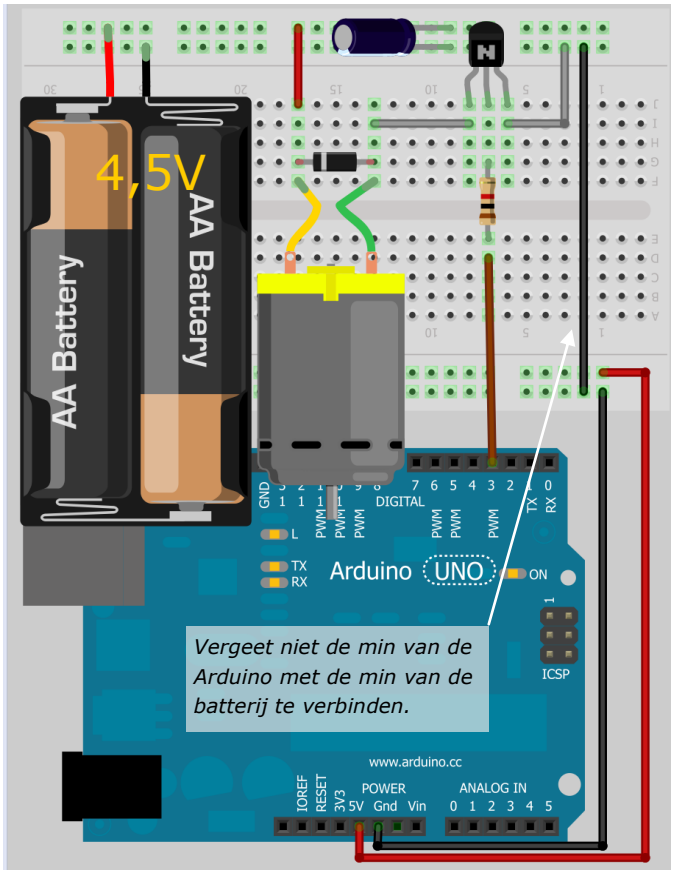
Wat gaan we doen?

De pin van een Arduino kan maximaal 40 milliampère leveren (zo'n twee brandende ledjes's). Dat is niet veel en als we 'stroomvreters' zoals motoren willen gebruiken moeten we dan ook gebruik maken van een externe voedingsbron (batterij) en een transistor. Een transistor is een soort schakelaartje dat werkt op elektrische stroom. Tussen c (collector) en e (emitter) kan alleen stroom lopen als er een beetje stroom door b (basis) loopt. Dat beetje stroom kan de Arduino pin best leveren en dan kan de stroom van de batterij door de motor en transistor lopen. Eerst gaan we het principe bekijken met alleen een led en transistor. Monteer de onderdelen zoals op de afbeelding rechts en upload de sketch **Les 5 transistor.** De collector wordt verbonden met de 5V van de Arduino en de emitter geeft de stroom door naar de led. De basis mag niet teveel stroom krijgen (anders gaat hij stuk) en wordt daarom via een 1 Kohm weerstand verbonden met pin 6. Als pin 6 nu hoog is laat de transistor stroom door en gaat de led branden. We gebruiken een iets aangepaste 'Blink' versie.




```
// Transistor (in feite het blink programma).
void setup() {
 pinMode(3, OUTPUT); // pin 6 verbonden met de
                     // basis van de transistor
}
void loop() {
  digitalWrite(3, HIGH); // led aan
  delay(2000);           // wacht 2 sec.
  digitalWrite(3, LOW);  // led uit
  delay(1000);           // wacht 1 sec.
}
```

Nu gaan we de motor aansluiten. De stroom die door de motor gaat komt van de batterij: Eén contact van de motor is verbonden met de +, het andere via de transistor met de min. Alleen de basis van de transistor is verbonden met de Arduino (pin3). Verder hebben we nog twee nieuwe onderdelen nodig die het circuit beschermen tegen overbelasting, vooral tijdens het starten en stoppen van de motor. Een *diode* wordt tussen de twee motoraansluitingen geplaatst. Let op de richting: Met de ring aan de plus kant!!. Een *condensator* wordt geplaatst tussen de plus en min van de batterij. Let ook hier goed op de plus en min: Lange pootje plus, min kort (staat ook op de behuizing). Monteer de onderdelen en upload **Les5 motor.** Met PWM kunnen we de motor ook harder en zachter laten draaien: Probeer **Les5 motor PWM** en **Les5 motor accelereren** (Uitleg in de sketch)

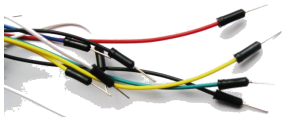


Les 6

Muziek Tone

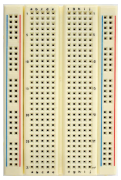
Wat heb je nodig?

R 100 Ω bruin/ zwart/bruin



Jumper wire

Speaker 8Ω



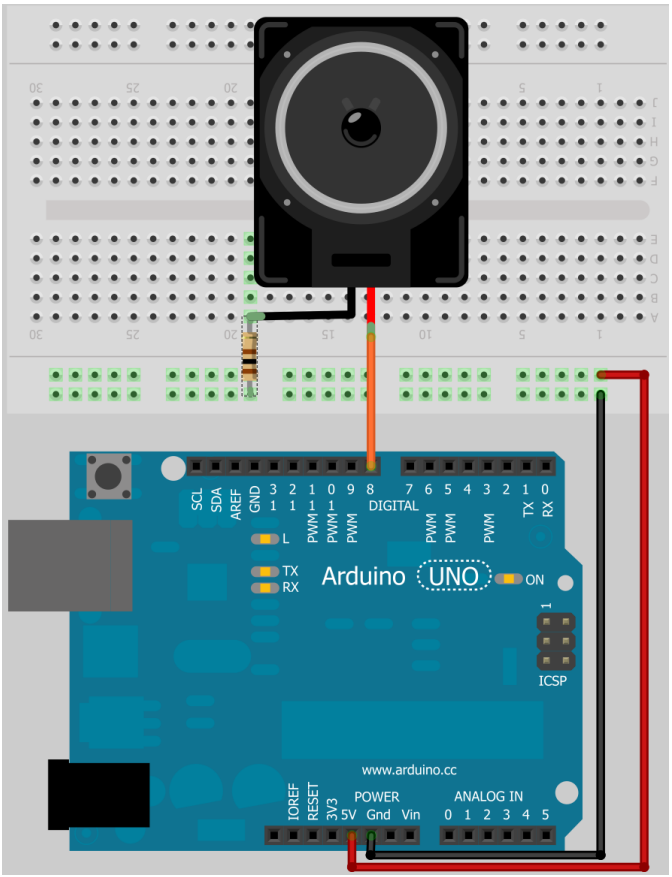
Breadboard

Arduino

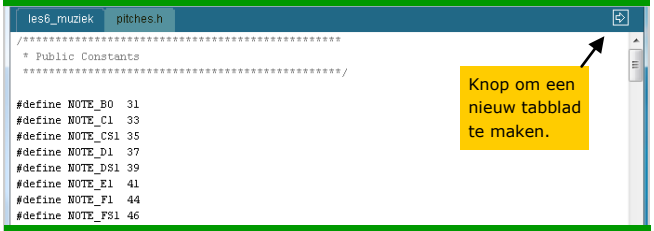
Wat gaan we doen?

Met de instructie *tone* kunnen we de Arduino tonen laten afspelen. De schrijfwijze is: tone (pinnummer, frequentie, tijdsduur). In de sketch hieronder is de speaker op pin 8, de frequentie van de eerste toon is 93 hertz, de duur van de toon is 1000 milliseconden. Je kan zo zoveel tonen achter elkaar plakken als je wilt. Upload **Les6 tone simpel** en probeer het uit. Je ziet dat alles nu in de setup staat en de loop is leeg. Zo wordt het geluid na uploaden maar één keer afgespeeld en niet eindeloos herhaald (het is geen Stradivarius).

```
// Tone simpel
int speakerpin = 8;

void setup (){
  //tone(pin, frequency, duration)
  tone(8, 93,1000);
  delay(1000);
  tone(8, 1047,1000);
}
void loop (){
}
```



Het kan ook een stuk slimmer. We kunnen van de melodie van een muziekstuk een verzameling noten maken: Een array (aangegeven met []). Dat is een collectie variabelen, in de sketch rechtsonder staat de array: int melody[] ={…….}. Daarna wordt de duur van de noot aangegeven (hier in 8e en kwart noten). Met de for loop worden alle 8 noten achtereenvolgens gespeeld (0 doet ook mee). Met tone wordt de eigenlijke instructie gegeven om spanning op de pin (8) te zetten (net als bij *tone simpel*). Bij tone simpel zagen we dat we alleen de frequentie van de noot konden invoeren. Om ook direct de noten te kunnen invoeren is een bestand gemaakt waarbij de noten aan een frequentie gekoppeld zijn. Dit wordt in een tweede tabblad (pitches.h) in de sketch geplaatst. De instructie *#include "pitches.h"* zorgt ervoor dat die informatie in de sketch ook gebruikt wordt.



Upload: **les6 muziek.** Voeg als je kopieert en plakt pitches.h toe: Maak een nieuw tabblad (zie afb. boven) noem het pitches.h en plak het bestand in de sketch. Verander de noten en de tijdsduur. Je kan een bestaand werk of je eigen muziekstuk maken.

les6_muziek | pitches.h

```
/*Melody. Plays a melody
 circuit:
 8-ohm speaker on digital pin 8
 created 21 Jan 2010, modified 30 Aug 2011 by Tom Igoe
 This example code is in the public domain.
 http://arduino.cc/en/Tutorial/Tone
*/
 #include "pitches.h"

// 8 notes in the melody (one is 0):
int melody[] = {
  NOTE_C4, NOTE_G3,NOTE_G3, NOTE_A3, NOTE_G3,0, NOTE_B3, NOTE_C4};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4,4,4,4,4 };

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 8; thisNote++) {

    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000/noteDurations[thisNote];
    // tone: (pin8...frequency (noteC4 etc. from pitches.h)...duration)
    tone(8, melody[thisNote],noteDuration);
    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    // stop the tone playing:
    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody. please.
}
```

Les 7

Relais

Wat heb je nodig?


DC Motor

Batterij 4,5 V


Diode 1N4001


1 KΩ bruin/zwart/rood

Relais


Led's rood+ groen+ 330 Ω 4x

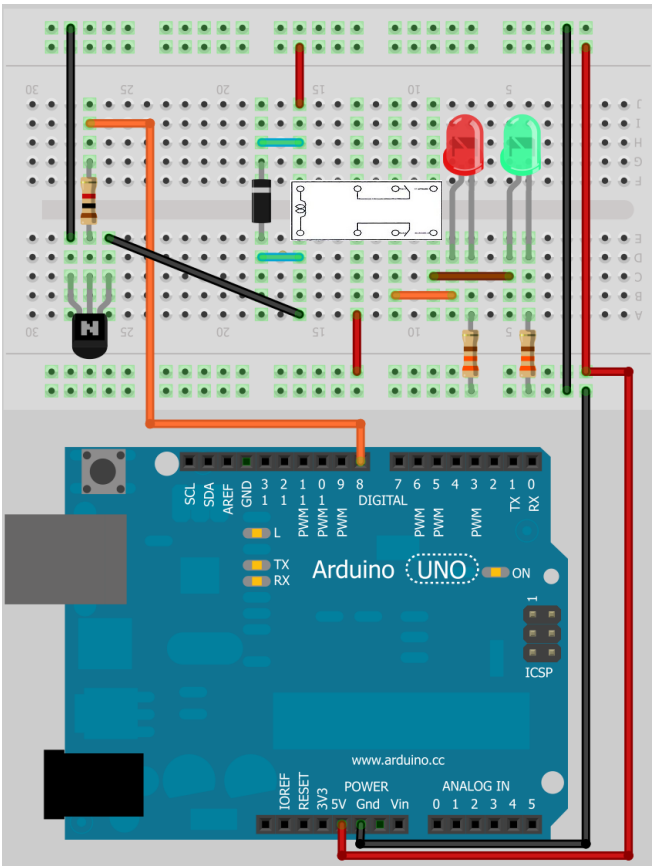
Transistor

Wat gaan we doen?

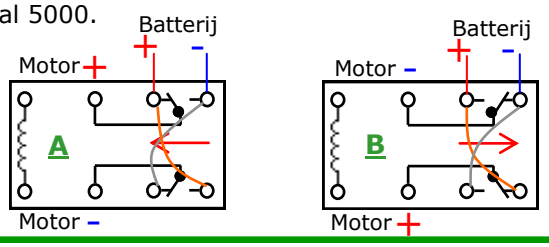
Als je iets op een arduino wil aansluiten dat veel stroom verbruikt kan dat niet zomaar. Dat hebben we al gezien bij les 3 (servo motor) en les 5 (dc motor). We hebben dan een 'schakelaar' nodig die door een pin van de arduino bediend kan worden waarmee een grotere stroom geschakeld kan worden. Een veel gebruikte 'schakelaar' hiervoor is een *relais.* Deze opent en sluit contacten d.m.v. een elektromagneet (de spiraal in het schema). Het relais dat wij gebruiken heeft twee standen. Als de elektromagneet bekrachtigt is, is een paar contacten verbonden. Staat er geen stroom op de magneet, dan zijn de anderen verbonden (zie onderstaand schema). Omdat ook een elektromagneet meestal veel stroom vraagt moet die bediend worden via een transistor zoals in les 5. Ook hier wordt weer een diode geplaatst (flyback diode) tussen de contacten van de elektromagneet (met de ring aan de plus kant!!). Eerst gaan we het relais uitproberen met ledjes, zo kan je goed zien welke contakten verbonden zijn.

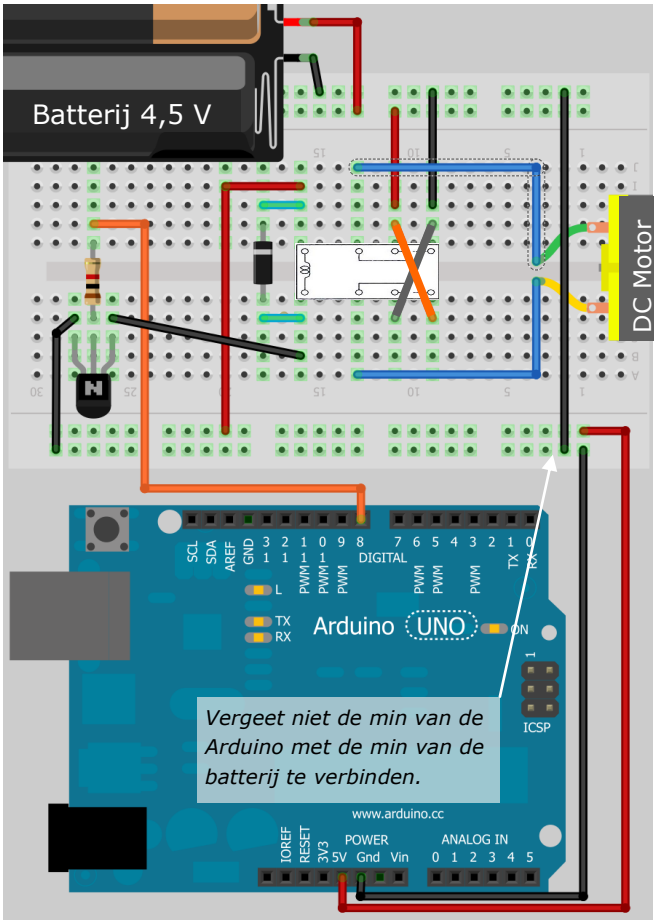
Monteer de onderdelen zoals in het schema rechtsboven en upload **File <Examples <1.Basic <Blink.** We gebruiken nu pin 8 en niet pin 13, dus je moet de sketch even aanpassen. Als alles goed is aangesloten hoor je de karakteristieke klik van het relais en knipperen de lampjes.

Opdracht: Sluit aan de andere kant van het relais nog een rode en groene led aan zodat die ook mee knipperen.



Nu gaan we een dc motor aansluiten op het relais. De stroom voor de motor komt van de batterij, net als bij les drie. Omdat ons relais dubbele contacten heeft kunnen we de draairichting van de motor veranderen. Als de plus en de min omgekeerd worden keert ook de draairichting om. Door twee schuin tegenover elkaar liggende contacten door te verbinden keert de richting van de stroom om als het relais al of niet bekrachtigd wordt (zie de schema's hieronder). Monteer alle onderdelen zoals op het schema hiernaast. Werk weer met de 'blink' versie van hierboven. Maak de *delay* minimaal 5000.





Les 8
Infrarood Sensor Obstakel Ontwijkend Object

Wat heb je nodig?


Speaker 8Ω


Bat. 4,5 V


Relais


Diode 1N4001


Sharp 2y0a21

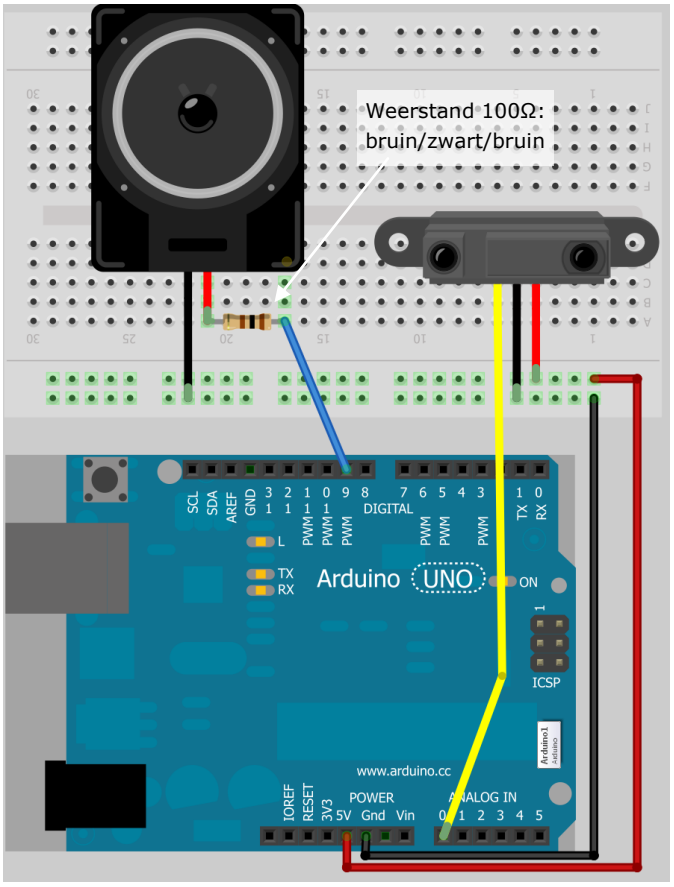

Weerstand: 100Ω en 1 KΩ

Transistor


Robot platvorm

Wat gaan we doen?

We gaan de Sharp 2y0a21 infrarood sensor uitproberen. Deze sensor straalt infrarood licht uit, vangt het teruggekaatste licht weer op en zet het om naar een spanning tussen +/- 3 en 0 volt, afhankelijk van de afstand. De Arduino maakt daar een getal van tussen 0 en 1023 dat we kunnen aflezen met de serial monitor (zie les 4). We kunnen het signaal ook hoorbaar maken door twee commentaarregels te 'ontslashen'. Monteer de onderdelen zoals in de afbeelding hiernaast en upload: **Les8 Sharp ir pin uitlezen.**



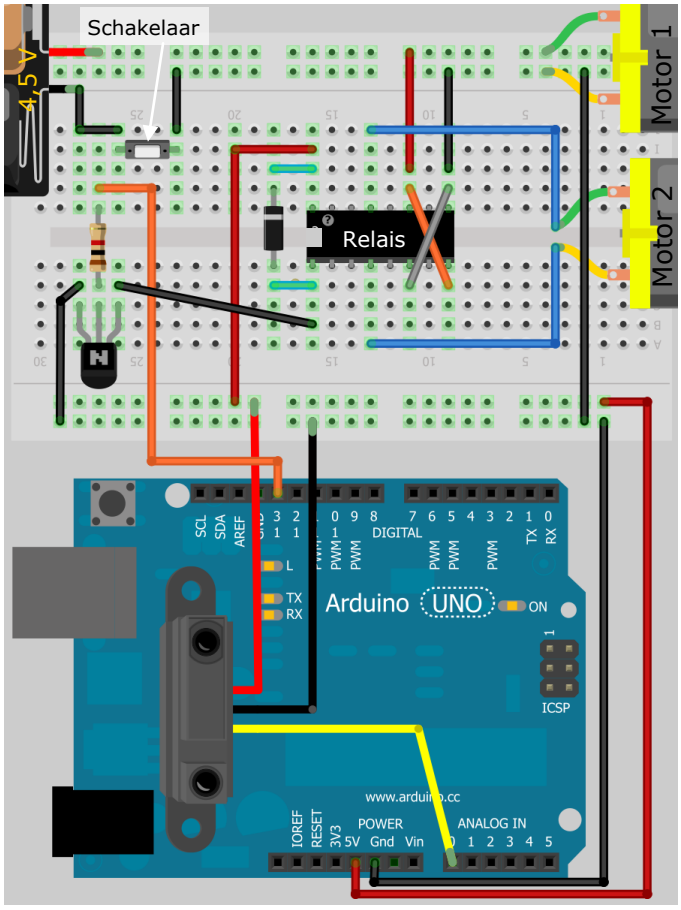

```
Les8 Sharp IR pin uitlezen
*/
int sensorpin = 0;       // IR sensor verbonden met analooge pin 0
int val = 0;             // variabele voor de analooge pin 0 waarde (start bij 0)
//int speakerPin = 9;    // speaker met een draad aan pin 9, da andere aan de min.
void setup()
{
  Serial.begin(9600);    // start de seriele monitor
}

void loop()
{
  //tone(speakerPin, val);      // maakt een toon met de waarde van val
  val = analogRead(sensorpin);  // leest de waarde van de IR sensor
  Serial.println(val);          // schrijft de waarde van de sensor naar de monitor
  delay(100);                   // wachttijd voor het schrijven van een nieuwe waarde
}
```

Nu krijgen we wel een getal op de monitor maar daarmee weten we nog niet de afstand. Omdat de relatie tussen de spanning en de afstand niet lineair is hebben we een wat ingewikkelde formule nodig: afstand=12343,85*pow(analogRead(IRpin, -1,15). Upload: **les8 Sharp IR naar cm.**

Opdracht: Vergelijk en noteer de waarde van de monitor met zelf gemeten waarde. Doe dit 10keer

```
// Les8 sharp IR naar cm.
// Omrekening Sharp 2y0a21 sensorwaarde naar centimeters
int IRpin = 0;          // IR sensor op analoog 0
void setup() {
  Serial.begin(9600);  // start de seriele monitor
}
void loop() {

  float distance =  12343.85 * pow(analogRead(IRpin),-1.15);
                          //12343.85 keer de waarde van pin 0 tot de macht -1.15
  Serial.print("Afstand in cm = ");              // print 'Afstand in = '
  Serial.println(distance);                      // print de afstand
  delay(100);                                    // wachttijd
}
```

We hebben nu de afstand in centimeters en dat is handig. We kunnen b.v. zeggen: Als de afstand kleiner is dan 30, dan moet pin 13 hoog zijn. Verbinden we nu pin 13 met een ompoolrelais (les 7), dan kunnen we één motor van het platvorm andersom laten draaien zodat het platvorm een bocht maakt als het ergens tegenaan dreigt te botsen. Monteer de onderdelen zoals in de afbeelding hiernaast op het platvorm en upload: **Les8 obstakel ontwijkend object.**




```
Les8 obstakel ontwijkend object.
*/
int IRpin = 0;                          // IR sensor op analoog 0
int relaisPin = 13;                     // pin 13 bedient het relais
void setup() {
pinMode (relaisPin, OUTPUT);
}
void loop() {
 float distance =  12343.85 * pow(analogRead(IRpin),-1.15);
  if (distance < 30) {                  // Als de afstand kleiner is dan 30 cm
    digitalWrite (relaisPin, HIGH);     // Relais hoog (platvorm draait)
  } else {                              // Is de afstand niet kleiner dan 30 cm
    digitalWrite (relaisPin, LOW);      // Relais laag (platvorm rijdt rechtdoor)
  }
  delay(300);                           // wachttijd (draaitijd)
}
```

Les 9

LDR Foto weer-stand

Wat heb je nodig?


Speaker 8Ω

Bat. 4,5 V


10 Kohm 5x


Weerstand: 100Ω


Servo motor

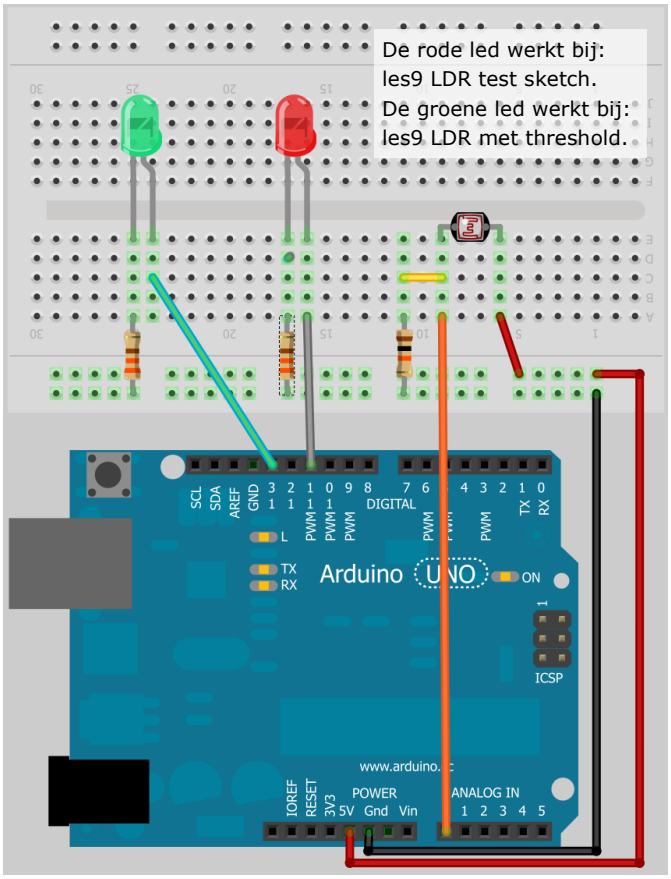

LDR 5x


Led rood+ groen + 2x 330 Ω

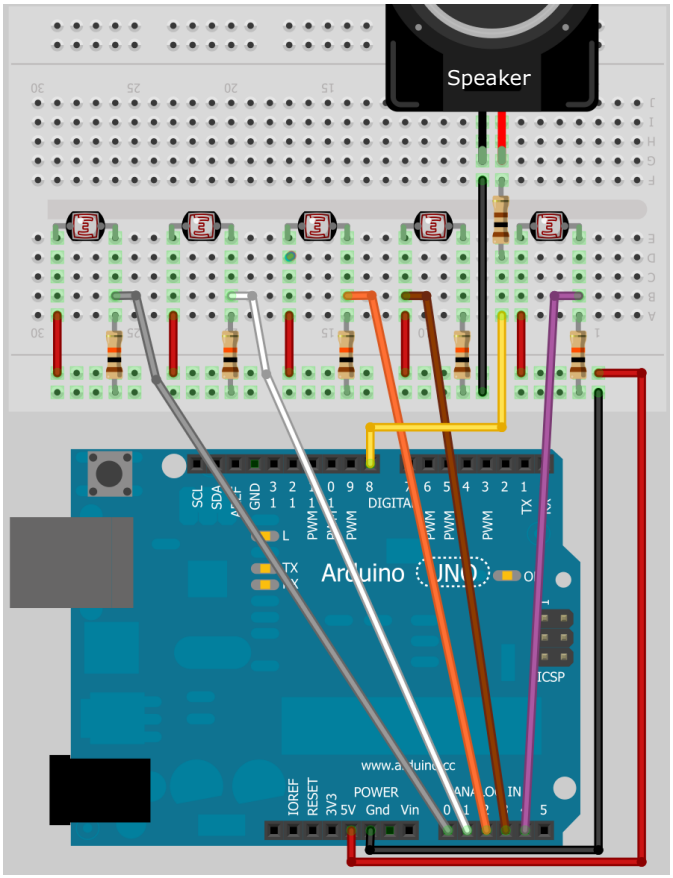
Wat gaan we doen?

Bij een LDR (light dependent resistor: licht afhankelijke weerstand) is de weerstand afhankelijk van de hoeveelheid licht die erop valt. Hoe meer licht, hoe minder weerstand. We gaan eerst de sensor uitproberen met een test programmaatje. Monteer de onderdelen zoals op de afbeelding hiernaast en upload: **les9 LDR test sketch.**
Lees vooral ook de toelichting bij de sketch.

```
Meer uitleg: www.ladyada.net/learn/sensors/cds.html
Les9 LDR test sketch.
*/
int ldrPin = 0;     // de LDR is verbonden met analoge pin 0
int ldrWaarde;      // De uitgeleZen waarde op de analoge pin 0
int LEDpin = 11;    // Rode led verbonden met pin 11 (PWM pin)
int LEDhelderheid;  // Helderheid van de led (tussen 0 en 255)

void setup(void) {
  Serial.begin(9600);   // Start de serial mmonitor
}
 void loop(void) {
  ldrWaarde = analogRead(ldrPin);
  Serial.print("Analog reading = ");
  Serial.println(ldrWaarde);  // print de ruwe analoge waarde op de monitor
  ldrWaarde = 1023 - ldrWaarde;                    // zie toelichting boven
  LEDhelderheid = map(ldrWaarde, 0, 1023, 0, 255); // zie toelichting boven
  analogWrite(LEDpin, LEDhelderheid);
 delay(100);
}
```



In plaats van de helderheid te variëren kunnen we een led bij een bepaalde hoeveelheid licht ook aan of uit laten gaan. Dat kan met de instructie: *Threshold.* We kunnen zeggen: Als de waarde op de ldr pin kleiner is dan… , dan moet pin … hoog. Zo kunnen we bij voorbeeld met transistor/relais schakelingen (les 7) zware lampen, motoren of pneumatiek aansturen. Upload: **Les9 LDR met threshold**. Test het programma en verander ook eens de threshold waarde.(N.B. Nu de groene led).
Opdracht suggestie: Maak een trommel, die werkt door het onderbreken van een lichtstraal, met behulp van een LDR en een servomotor (les 3). Een servo is eigenlijk wat langzaam, probeer het effect te vergroten (Langere stok? Veer?). Je kan er ook een solenoid bijzetten als tweede drum (zie pag. 25). Probeer de sketch zelf te maken. Lukt het niet? Upload: les9 ldr trommel.

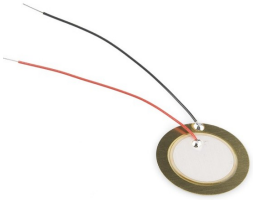


Met lichtsensoren zijn veel leuke dingen te doen. In het laatste voorbeeld maken we een 'pentatonische toonladder' (die wordt vaker in bluesmuziek gebruikt en kost maar vijf sensoren). De toonladder is CDEGA en elke toon is gekoppeld aan een sensor (en analoge pin). De sensoren worden achtereenvolgens met een *for loop* 'uitgelezen'. Monteer de onderdelen zoals op de afbeelding hiernaast en upload: **les9 LDR blues.**
Voor meer info zie de toelichting in de sketch.

Tot nu toe hebben we een aantal 'sensoren en actuatoren' besproken en gebruikt. Er zijn er natuurlijk nog veel meer. Op de volgende pagina's zien we er nog een aantal. Zij zitten niet in iedere 'kit' maar ze zijn wel beschikbaar en gemakkelijk om te gebruiken.
Bij ieder onderdeel staat een verwijzing naar een (Engelstalige) handleiding of tutorial.

Temperatuur sensor:
De behuizing is hetzelfde als een transistor, dus vergis je niet.
Temperatuursensor DS18B20:
Tutorial: http://bildr.org/2011/07/ds18b20-arduino/
Temperatuursensor TMP 36:
Tutorial: http://www.ladyada.net/learn/sensors/tmp36.html
Tutorial: http://oomlout.com/TMP36/TMP36-Guide.pdf

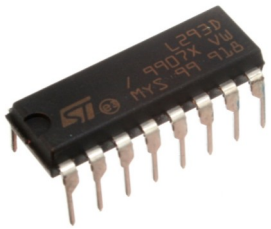
Piezo druk, geluid sensor:
Een piezo element zet een schok om in elektriciteit en andersom.
Tutorial: http://www.arduino.cc/en/Tutorial/Knock



Tilt sensor:
Eenvoudige sensor die een contact maakt of verbreekt bij kantelen.
Tutorial: http://www.ladyada.net/learn/sensors/tilt.html

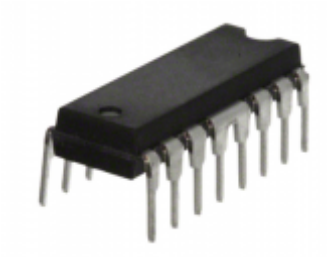
Ping ultrasoon sensor:
Afstandsensor die werkt met ultrasoon geluid.
Handleiding: http://www.parallax.com/Portals/0/Downloads/docs/prod/acc/28015-PING-v1.6.pdf
Tutorial: http://arduino.cc/en/Tutorial/Ping

Motor Driver L293D:
'H-brug' IC om DC motoren aan te sturen.
Tutorial: http://itp.nyu.edu/physcomp/Labs/DCMotorControl
Quick-start: http://oomlout.com/L293/L293-Guide.pdf
Datasheet: http://oomlout.com/L293/IC-L293D-DATA.pdf

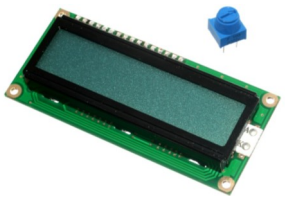


Pir: Passive IR sensor
Infrarood sensor die beweging kan dedecteren met warmte-verschillen.
Tutorial: http://www.ladyada.net/learn/sensors/pir.html
Tutorial: http://bildr.org/2011/06/pir_arduino/
Datasheet: http://www.sparkfun.com/datasheets/Sensors/Proximity/SE-10.pdf

Shift Register 74HC595:
Hiermee kan je het aantal pinnen uitbreiden. Je gebruikt drie pinnen en je krijgt er acht bij. Leuk voor bij voorbeeld veel led's.
Tutorial: http://www.instructables.com/id/Arduino-Expermentation-Kit-How-to-get-Started-wi/step7/8-More-Leds-74HC595-Shift-Register-CIRC05/
Datasheet: http://www.oomlout.com/ARDX/ZZ-DATA/74HC595.pdf



LCD Scherm:
Twee regelig LCD scherm.
Quick-start: http://oomlout.com/LCDD/LCDD-Guide.pdf
Tutorial: http://arduino.cc/en/Tutorial/LiquidCrystal
Datasheet: http://oomlout.com/LCDD/LCDD-SUMM-BC1602A.pdf



Solenoid (elektromagneet):
Elektromagneet: Als je er spanning op zet gaat een as op en neer. Handig voor korte lineaire bewegingen.
Datasheet: http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Robotics/ZHO-420S.pdf
Aansluit schema:http://www.arduino.cc/playground/uploads/Learning/solenoid_driver.pdf

?

Bij het samenstellen van deze introductie is naast eigen materiaal vooral gebruik gemaakt van de informatie op de Arduino site:
http://www.arduino.cc/
Ook van de site van 'ladyada' is dankbaar gebruik gemaakt, vooral les 1c:
http://www.ladyada.net/learn/arduino/
En de ARDX Experimeter's Guide:
http://oomlout.com/a/products/ardx/

De schema's zijn gemaakt met het open source programma Fritzing:
http://fritzing.org/

Leveranciers in Nederland voor Arduino en componenten:
http://www.pieterfloris.nl/shop/
http://iprototype.nl/

Deze Nederlandstalige *Interactief Ontwerpen met Arduino introductie* is gemaakt omdat leerlingen uit de eerste jaren van het secundair onderwijs moeilijk uit de voeten konden met het bestaande Engelstalige materiaal. Dat is ook niet verwonderlijk omdat zowel het Engels als de computertaal (Een variant van C++, Java) in feite vreemde talen zijn die tegelijkertijd en door elkaar aangeleerd moesten worden. Door de instructie in de moedertaal te geven is dit probleem verholpen.
Een uitgangspunt is verder dat het werken met Arduino en programmeren geen doel op zich is, maar bedoeld om zo snel en gemakkelijk mogelijk projecten te kunnen realiseren ook zonder voorkennis van programmeren en elektronica.
Daarnaast is belangrijk dat met het materiaal zelfstandig, zonder veel instructie, gewerkt kan worden.

Testversie