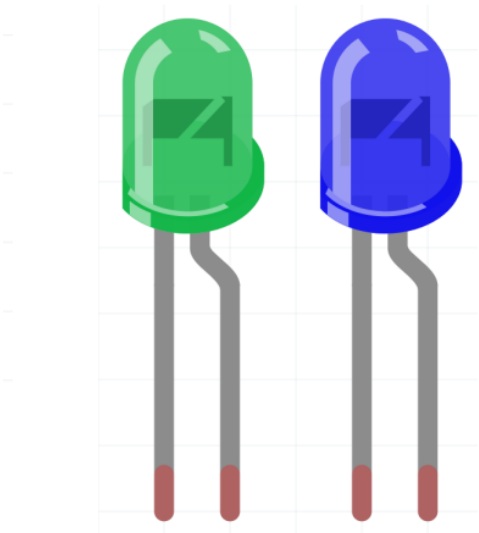

DE JONGE ONDERZOEKERS
Kijk, doe, ontdek !

Les 5: Blink 6
Les 6: Knop kiest LED
Les 7: Potmeter kiest LED
Les 8: LDR kiest LED

Figure 1: Boek 2: LEDs

Contents

Voorwoord

Dit is een boek over Arduino, geschreven voor jonge tieners. Een Arduino is een machine die je kunt programmeren. Dit boek leert je hoe je elektronica op de Arduino aansluit, en hoe je deze programmeert.

Over dit boek

Dit boek heeft een CC-BY-NC-SA licensie.

Figure 1: De licensie van dit boek

Het is nog een beetje een slordig boek. Er zitten tiepvauten in en de opmaak is niet `altijd` *even mooi*. Omdat dit boek op een website staat, kan iedereen die dit boek te slordig vindt minder slordig maken.

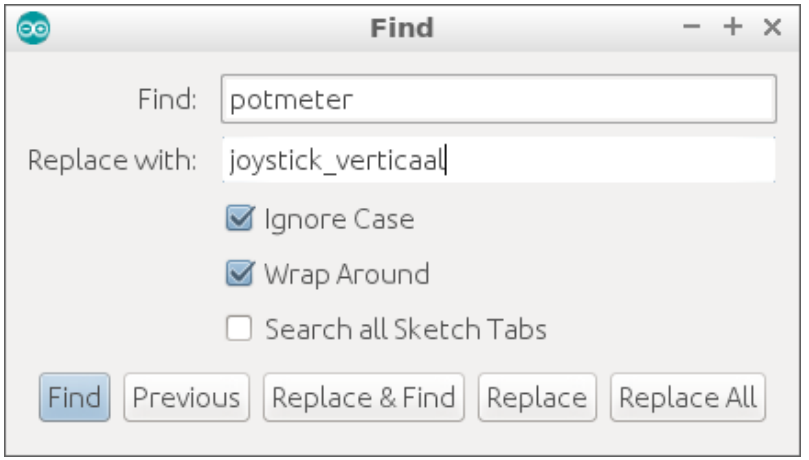

Figure 6: Find, klik hier op 'Replace All'

```
int lees_joystick_verticaal()
{
  return analogRead(pin_joystick_verticaal);
}

void laat_joystick_verticaal_zien()
{
  Serial.print(lees_joystick_verticaal());
  analogWrite(pin_led, lees_joystick_verticaal() / 4);
}

void loop()
{
  laat_joystick_verticaal_zien();
  // ...
}
```

7.12 Potmeter: eindopdracht

- Sluit een tweede LEDje aan
- Dit tweede LEDje moet reageren zoals het eerste LEDje, maar dan als de joystick horizontaal wordt bewogen

```
  analogWrite(pin_led, lees_potmeter());
}
```

Als je aan de potmeter draait, zie je dat het LEDje vier keer vloeiend aan gaat.

7.9 Potmeter: goed sturen, opdracht

- Verander de volgende code ...

```
analogWrite(pin_led, lees_potmeter());
```

... naar dit:

```
analogWrite(pin_led, lees_potmeter() / 4);
```

- Wat zie je?
- Wat denk je dat `/` betekent? Tip: waar zie je dit soort strepen bij rekenen?

7.10 Potmeter: goed sturen, oplossing

- Je ziet dat het LEDje nu mooi van uit naar aan gaat als je aan de potmeter draait
- De `/` betekent 'gedeeld door'. Dit is dezelfde deelstreep als bij breuken en procenten!

7.11 Potmeter: joystick aansluiten, opdracht

- Vervang de potmeter door een joystick. Leg de volgende verbindingen:

Joystick	Arduino
VCC	5V
V	A0
H	A1
GND	GND

- Als je dit goed hebt aangesloten, kun je met de joystick nu het LEDje besturen

7.11 Potmeter: joystick aansluiten, oplossing

[stroomschema]

7.12 Potmeter: joystick lezen, opdracht

In de code, vervang de tekst `potmeter` door `joystick_verticaal` Gebruik hiervoor 'Find' (CTRL-F of `Edit | Find`) en gebruik 'Replace All' ('Vervang alles').

7.12 Potmeter: joystick lezen, oplossing

```
const int pin_joystick_verticaal = A0;
// ...

void setup()
{
  pinMode(pin_joystick_verticaal, INPUT);
  // ...
}
```


Les 5: Blink 4

In deze les gebruiken we 4 LEDs en functies

5.1 Blink 6: Opdracht

- Maak een schakeling met 2 LEDs, elk met een weerstand van 1000 Ohm in serie
- Sluit de 1e LED aan op pin 2
- Sluit de 2e LED aan op pin 3
- Upload deze code:

```
const int pin_led_1 = 2;
const int pin_led_2 = 3;
const int wachttijd = 1000;

void setup()
{
  pinMode(pin_led_1, OUTPUT);
  pinMode(pin_led_2, OUTPUT);
}

void loop()
{
  digitalWrite(pin_led_1, HIGH);
  digitalWrite(pin_led_2, LOW);
  delay(wachttijd);
  digitalWrite(pin_led_1, LOW);
  digitalWrite(pin_led_2, HIGH);
  delay(wachttijd);
}
```

5.2 Blink 6: Oplossing

5.3 Blink 6: `effe_wachten`, opdracht

We gaan onze eerste functie schrijven!

- Voeg aan je code toe, boven `loop`:

```
void effe_wachten()
{
  delay(wachttijd);
}
```

- In `loop`, vervang twee keer `delay(wachttijd);` door `effe_wachten();`

5.4 Blink 6: `effe_wachten`, oplossing

```
// ...

void setup()
{
  // ...
}

void effe_wachten()
{
  delay(wachttijd);
```

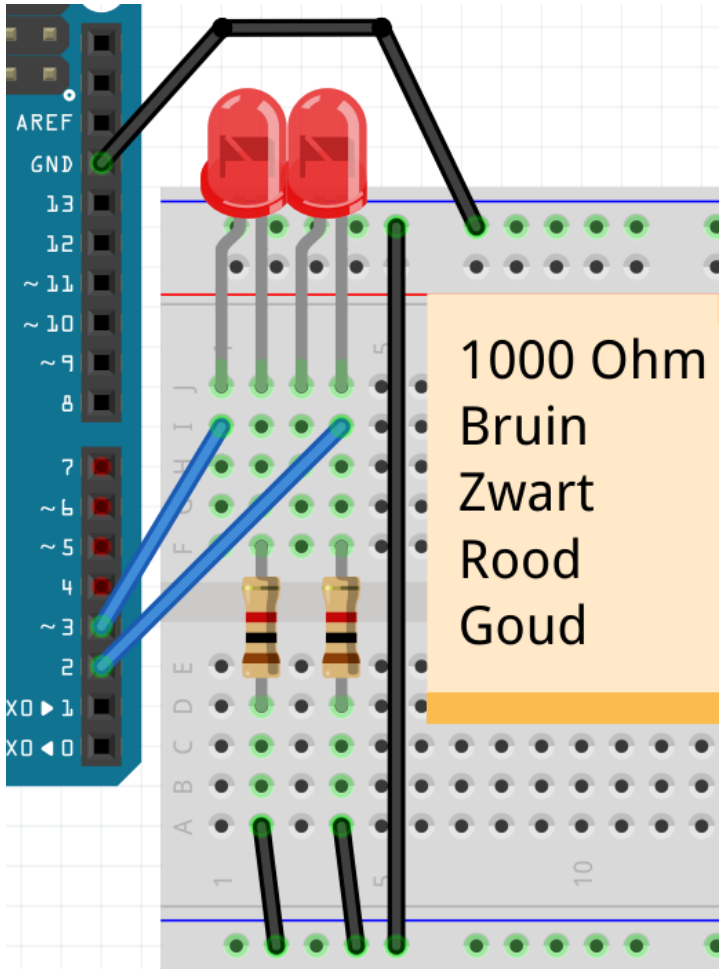

Figure 2: 5.2 Blink 6: Oplossing

- Voeg deze functie toe:

```
int lees_potmeter()
{
  return analogRead(pin_potmeter);
}
```

- In `laat_potmeter_zien` vervang de tekst `"potmeter"` door `lees_potmeter()`
- Upload het programma en draai aan de potmeter. Welke getallen komen uit?

7.6 Potmeter: lezen, oplossing

```
const int pin_potmeter = A0;
//...

void setup()
{
  // ...
  pinMode(pin_potmeter, INPUT);
}

int lees_potmeter()
{
  return analogRead(pin_potmeter);
}

void laat_potmeter_zien()
{
  Serial.print(lees_potmeter());
}
```

De getallen die uit `lees_potmeter` komen zitten tussen nul en 1024 in.

7.7 Potmeter: sturen, opdracht

- Sluit een LED aan op pin 11
- Maak een variabele `pin_led` met de juiste waarde
- In `setup`, zet de `pinMode` van `pin_led` op `OUTPUT`
- In `laat_potmeter_zien` voeg deze regel toe:

```
analogWrite(pin_led, lees_potmeter());
```

- Upload en draai aan de potmeter. Wat zie je?

7.8 Potmeter: sturen, oplossing

```
// ...
const int pin_led = 11;

void setup()
{
  // ...
  pinMode(pin_led, OUTPUT);
}

void laat_potmeter_zien()
{
  // ...
```

Les 7: Potmeter en joystick

In deze les gebruiken we een potmeter, een joystick en een functie die een waarde teruggeeft.

7.1 Potmeter: aansluiten, opdracht

- Sluit een potmeter aan. Leg de volgende verbindingen:

Potmeter	Arduino
Linker pootje	5V
Middelste pootje	A0
Rechter pootje	GND

7.2 Potmeter: aansluiten, oplossing

[schema hier]

7.3 Potmeter: opstarten, opdracht

- In `setup`, start de seriele monitor op 9600 baud
- Maak een variabele `wachttijd` met een waarde van 100
- Maak een functie `effe_wachten` die het programma `wachttijd` milliseconde laat wachten
- Maak een functie `laat_potmeter_zien`, die het woord `potmeter` naar de seriele monitor stuurt
- In `loop`, gebruik `laat_potmeter_zien` en `effe_wachten`

7.4 Potmeter: opstarten, oplossing

```
const int wachttijd = 100; //milliseconden

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

void laat_potmeter_zien()
{
  Serial.print("potmeter");
}

void effe_wachten()
{
  delay(wachttijd);
}

void loop()
{
  laat_potmeter_zien();
  effe_wachten();
}
```

7.5 Potmeter: lezen, opdracht

- Maak een variabele `pin_potmeter` met als waarde `A0`.
- In `setup`, zet de `pinMode` van `pin_potmeter` op `INPUT`


```
}

void loop()
{
  // ... [zet alleen LED 1 aan]
  effe_wachten();
  // ... [zet alleen LED 2 aan]
  effe_wachten();
}
```

5.5 Blink 6: `zet_alleen_led_1_aan`, opdracht

- Schrijf een functie, `zet_alleen_led_1_aan`, die ervoor zorgt dat alleen LED 1 brandt (oftewel: LED 2 moet uit)
- Gebruik `zet_alleen_led_1_aan` in `loop`

5.6 Blink 6: `zet_alleen_led_1_aan`, oplossing

```
void zet_alleen_led_1_aan()
{
  digitalWrite(pin_led_1, HIGH);
  digitalWrite(pin_led_2, LOW);
}

void loop()
{
  zet_alleen_led_1_aan();
  // ...
}
```

5.7 Blink 6: `zet_alleen_led_2_aan`, opdracht

- Schrijf een functie, `zet_alleen_led_2_aan`, die ervoor zorgt dat alleen LED 2 brandt (oftewel: LED 1 moet uit)
- Gebruik `zet_alleen_led_2_aan` in `loop`

5.8 Blink 6: `zet_alleen_led_2_aan`, oplossing

```
void zet_alleen_led_2_aan()
{
  digitalWrite(pin_led_1, LOW);
  digitalWrite(pin_led_2, HIGH);
}

void loop()
{
  // ...
  zet_alleen_led_2_aan();
}
```

5.7 Blink 6: `zet_alleen_led_3_aan`, opdracht

- Sluit een derde LEDje aan, op pin 4
- Maak een nieuwe variabele `pin_led_3` voor deze LED
- Schrijf een functie, `zet_alleen_led_3_aan`, die ervoor zorgt dat alleen LED 3 brandt (oftewel: LEDs 1 en 2 moeten uit)
- Gebruik `zet_alleen_led_3_aan` in `loop`

- In `loop`, laat eerst alleen LED 1 branden, wacht effe, laat eerst alleen LED 2 branden, wacht effe, laat eerst alleen LED 3 branden, wacht effe

5.8 Blink 6: `zet_alleen_led_3_aan`, oplossing

```
// ...
const int pin_led_3 = 4;

void setup()
{
  // ...
  pinMode(pin_led_3, OUTPUT);
}

void zet_alleen_led_1_aan()
{
  // ...
  digitalWrite(pin_led_3, LOW);
}

void zet_alleen_led_2_aan()
{
  // ...
  digitalWrite(pin_led_3, LOW);
}

void zet_alleen_led_3_aan()
{
  digitalWrite(pin_led_1, LOW);
  digitalWrite(pin_led_2, LOW);
  digitalWrite(pin_led_3, HIGH);
}

void loop()
{
  // ...
  zet_alleen_led_3_aan();
  effe_wachten();
}
```

5.9 Blink 6: eindopdracht

- Sluit een vierde LEDje aan, op pin 5
- Maak een nieuwe variabele `pin_led_4` voor deze LED
- Schrijf een functie, `zet_alleen_led_4_aan`, die ervoor zorgt dat alleen LED 4 brandt (oftewel: LEDs 1 en 2 en 3 moeten uit)
- Gebruik `zet_alleen_led_4_aan` in `loop`
- In `loop`, maak een Nightrider patroon: laat omstebeurt branden LEDs 1, 2, 3, 4, 3, 2, 1. Steeds ertussen even wachten

6.12 Oplossing

```
void reageer_op_knop()
{
  if (digitalRead(pin_knop) == HIGH)
  {
    aantal = aantal + 1;
  }
}

void loop()
{
  // ...
  reageer_op_knop();
  // ...
}
```

6.13: Eindopdracht

- In `reageer_op_knop`: als de knop is losgelated, wordt `aantal` weer nul
- Sluit een LED aan op pin 13
- De LED brandt alleen als `aantal` meer is dan tien. Gebruik dit `if` statement:

```
if (aantal > 10)
{
  // ...
}
```

6.8 Oplaadknop: knop los, oplossing

```
const int wachttijd = 100;
// ... [variabele pin_knop]

// ...

void laat_knop_zien()
{
  if (/* de knop is ingedrukt */)
  {
    // ... [zeg dat de knop is ingedrukt]
  }
  else
  {
    Serial.println("Knop is niet ingedrukt");
  }
}
```

6.9 Oplaadknop: `aantal`, opdracht

- Maak een variabele `aantal`. Dit is een heel getal dat kan veranderen, met beginwaarde nul
- Maak een nieuwe functie, `laat_aantal_zien`. In deze functie wordt de waarde van `aantal` naar de seriele monitor gestuurd. Dit programmeer je met:

```
Serial.println(aantal);
```

- Gebruik `laat_knop_zien`, dan `laat_aantal_zien` en dan `wacht_effe` in `loop`

6.10 Oplaadknop: `aantal`, oplossing

```
// ...
int aantal = 0;

void laat_aantal_zien()
{
  Serial.println(aantal);
}

void loop()
{
  // ...
  laat_aantal_zien();
  // ...
}
```

6.11 Oplaadknop: `reageer_op_knop`, opdracht

- Maak een nieuwe functie, `reageer_op_knop`. In `reageer_op_knop`: als de knop is ingedrukt, wordt `aantal` 1 meer. Dit programmeer je met:

```
aantal = aantal + 1;
```

- Gebruik `reageer_op_knop` tussen `laat_knop_zien` en `laat_aantal_zien` in `loop`

Les 6: Oplaadknop

In deze les gebruiken we een knop, LEDs en een functie die een waarde teruggeeft.

6.1 Oplaadknop: Hoi, opdracht

- Je hoeft niks aan te sluiten!
- Upload deze code:

```
const int wachttijd = 1000;

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

void loop()
{
  Serial.println("Hoi");
  delay(wachttijd);
}
```

- Na het uploaden, klik op 'Serial Monitor'

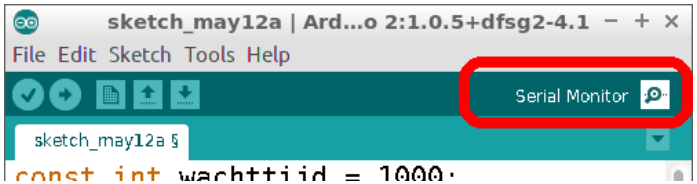

Figure 3: Klik op 'Serial Monitor'

Wat zie je?

6.2 Oplaadknop: Hoi, oplossing

Je ziet dat de Arduino 'Hoi' zegt!

6.3 Oplaadknop: `wacht_effe` en `laat_knop_zien`, opdracht

- Schrijf een functie `wacht_effe`: in deze functie wacht de Arduino `wachttijd` milliseconden
- Schrijf een functie `laat_knop_zien`: in deze functie zegt de Arduino (nu nog) 'Hoi'
- Gebruik `laat_knop_zien` en dan `wacht_effe` in `loop`

6.4 Oplaadknop: `wacht_effe` en `laat_knop_zien`, oplossing

```
// ...

void setup()
{
  // ...
}

void laat_knop_zien()
```

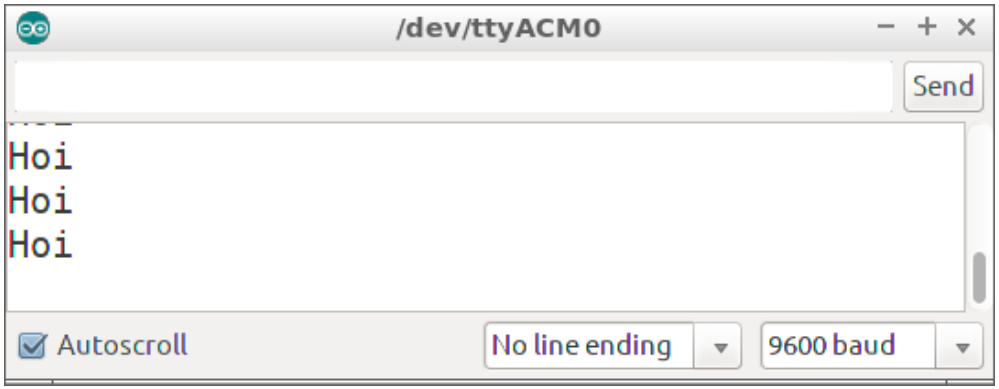

Figure 4:

```
{
  Serial.println("Hoi");
}


void wacht_effe()
{
  delay(wachttijd);
}

void loop()
{
  laat_knop_zien();
  wacht_effe();
}
```

6.5 Oplaadknop: knop, opdracht

- Sluit een knop aan op pin 2
- Maak een variabele `pin_knop`
- In `setup`, zeg met `pinMode` dat `pin_knop` een `INPUT` is
- Vervang `laat_knop_zien` door deze code:

```
void laat_knop_zien()
{
  if (digitalRead(pin_knop) == HIGH)
  {
    Serial.println("Knop is ingedrukt");
  }
}
```

6.6 Oplaadknop: knop, oplossing

```
// ...
const int pin_knop = 2;
```


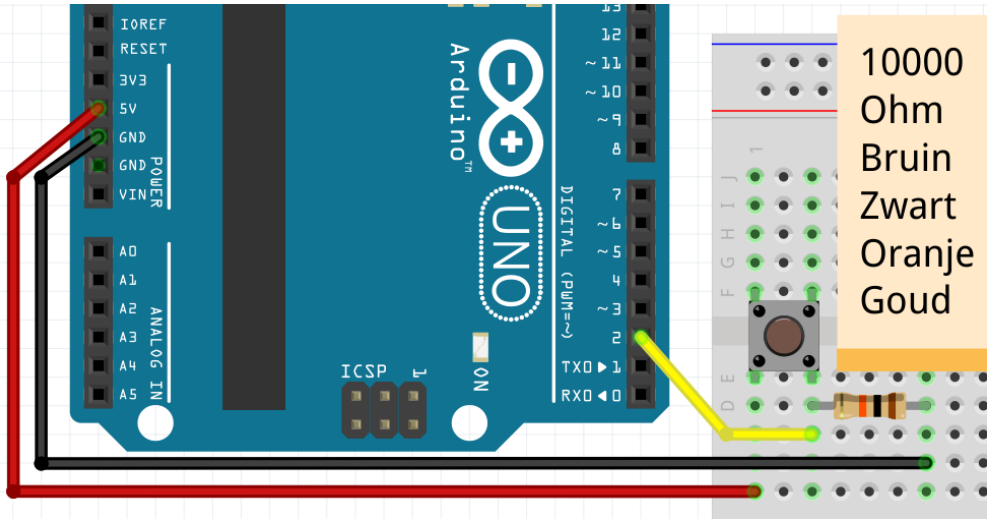

Figure 5: 6.6 Oplaadknop: knop, oplossing

```
void setup()
{
  // ...
  pinMode(pin_knop, INPUT);
}

void setup()
{
  // ...
}

void laat_knop_zien()
{
  if (digitalRead(pin_knop) == HIGH)
  {
    Serial.println("Knop is ingedrukt");
  }
}

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

6.7 Oplaadknop: knop los, opdracht

- In `laat_knop_zien`, als de knop niet is ingedrukt, laat de Arduino dan 'Knop is niet ingedrukt' zeggen
- Verander `wachttijd` naar 100 milliseconden