Zelf functies schrijven

Wat zijn Functies en Waarom hebben we ze nodig?

Een **functie** kunnen we zien als een groep instructies, die een specifieke taak uitvoert.

We maken een functie om ons programma beter georganiseerd te houden, beter leesbaar te maken, en vaak omdat we deze “taak” willen hergebruiken.

Standaard hebben alle Arduino scripts twee functie’s: Loop en Setup. Hieruit kunnen we al aflezen hoe een functie eruitziet, **dit is hoe je een functie declareert (ofwel maakt)**:

```
void setup() {
  // code
}
```

Woord voor woord:

- void: laat dit maar even voor wat het is: een functie begint met void.
- setup: de naam van de functie. Dit moet zonder spatie’s
- (): hiermee geef je aan dat het een functie is.
- {: hiermee geef je aan de je de functie opent.
- // code: dit is een comment, alles wat begint met // wordt niet uitgevoerd.
- }: hiermee geef je aan de je de functie sluit.

Alles binnen { en } is staat voor wat de functie moet doen.

Bij Arduino mag je alle functie’s maken die je wilt, je mag ze noemen hoe je wilt (je mag niet twee dezelfde namen hebben).
Een functienaam:

- Moet beginnen met een letter (a, b, …, z, A, B, … , Z) of een underscore (_)
- Mag letters bevatten
- Mag underscore(s) bevatten
- Mag nummers bevatten
- MAG GEEN speciale tekens, symbolen of spaties bevatten, zoals !@#$%^&*()+
- Is hoofdletter gevoelig!

Functie’s hebben drie gouden regels:

- Je kan alle functie’s maken die je wilt, maar je **moet** de functie’s setup en loop maken in je code (eis van Arduino).
- Je declareert (maakt, zie hierboven) functie’s nooit in een andere functie .
- Een functie wordt pas uitgevoerd als je hem uitvoert. De enige uitzonderingen hierop zijn loop en setup, deze voert Arduino zelf uit.

In deze les gaan we zelf een functie schrijven.

Functie voor morse-code.

Stel we willen een programma maken dat tekst kan omzetten in morse-code. Dan kunnen functies erg makkelijk zijn. We hoeven dan maar 1 keer de morse-code voor een letter te programmeren in een functie. En elke keer als we de functie aanroepen wordt automatisch de juiste code gemaakt.

Schakeling

We houden de schakeling simpel, want het gaat nu even om onze eigen zelfgeschreven functie(s). We gebruiken het kleine ledje op het Arduinbord om de morse-code uit te zenden. Het kleine ledje zit op pin 13.

Zelf functies schrijven

Code:

```
int pin_out = 13;

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

void loop() {
  morse_s(pin_out);
  delay(300);
  morse_o(pin_out);
  delay(300);
  morse_s(pin_out);
  delay(2000);
}

void morse_s(int pin) {
  digitalWrite(pin, HIGH);
  delay(100);
  digitalWrite(pin, LOW);
  delay(100);
  digitalWrite(pin, HIGH);
  delay(100);
  digitalWrite(pin, LOW);
  delay(100);
  digitalWrite(pin, HIGH);
  delay(100);
  digitalWrite(pin, LOW);
  delay(100);
}

void morse_o(int pin) {
  digitalWrite(pin, HIGH);
  delay(300);
  digitalWrite(pin, LOW);
  delay(100);
  digitalWrite(pin, HIGH);
  delay(300);
  digitalWrite(pin, LOW);
  delay(100);
  digitalWrite(pin, HIGH);
  delay(300);
  digitalWrite(pin, LOW);
  delay(100);
}
```

Zoals je ziet is onze code nu **duidelijk en goed leesbaar**. We hebben nu de programmeertaal uitgebreid, voor ons programma, met de functie “morse_o” en “morse_s”.

Maar functies hebben nog een groot voordeel. Stel we moeten de code voor een letter aanpassen, bijvoorbeeld omdat er een fout in morse_s zit. Bij een functie hoeven we dat dus maar op 1 plaats te doen: in de functie definitie. Onze code wordt dus **beter beheerbaar**.

Meer informatie is hier te vinden: https://www.tweaking4all.nl/hardware/arduino/arduino-programmeer-cursus/arduino-programmeren-deel-6/