Installatie en testen

Arduino-experiment	130101-EN	*Trefwoorden*	Installatie, keuze van board en port, init en loop, ingebouwde LED	
Versie	2018-06-26 / HS	*Niveau*	Basiscursus, module 1	*p.* 1/7

Wat je leert:

- Installeer het Arduino systeem op je PC
- Test de installatie
- Schrijf je allereerste Arduino programma
- Leer om programma's op te slaan en te openen

Deze module wijkt af van de rest: hij is iets langer dan de andere modules en bevat veel instructies en uitleg – maar relatief weinig uitdagingen. Hou vol!

Als het Arduino systeem al is geïnstalleerd op de PC die je gebruikt, spring dan direct naar onderdeel 2.

1 – Installatie

Download de Arduino IDE

Het Arduino systeem bestaat uit zowel hardware als software. De hardware – het Arduino bord – is programmeerbaar via een USB-kabel, verbonden met de PC.

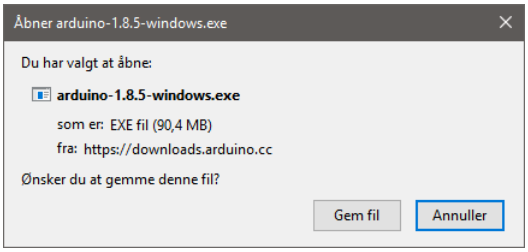
Programma's voor de Arduino zijn geschreven in een PC-programma, genaamd de *IDE*, wat staat voor "Integrated Development Environment". Klinkt technisch, maar het betekent gewoon dat het "het hele pakket" bevat dat nodig is voor

1. Schrijven, bewerken, opslaan en openen van programma's
2. Vertaling naar machinecode
3. Uploaden naar de Arduino hardware

Download de Arduino IDE op de website https://www.arduino.cc/en/Main/Software

Daar vind je de nieuwste versie – voor verschillende besturingssystemen. Kies de bovenste optie (hier met rood gemarkeerd) als je een Windows-PC hebt. (In de volgende beschrijving nemen we aan dat dit het geval is.)

De volgende pagina biedt de mogelijkheid om geld te doneren aan het Arduino-project en zo bij te dragen aan de verdere ontwikkeling van het systeem – of om het programma gewoon te downloaden.



Windows zal nu vragen wat je wilt doen met het bestand dat je downloadt – bewaar het ergens waar je het makkelijk terug kunt vinden.

Installatie

Na het downloaden van het installatiebestand start je de installatie door te dubbelklikken op het bestand.

Je zult worden gevraagd of je deze app toestaat om wijzigingen aan te brengen op je apparaat – dat is wat je wilt, dus: “Ja”.

In de volgende dialoogvensters selecteer je gewoon de standaardopties.

Als de installaties klaar zijn, kun je de Arduino IDE openen met behulp van het icoontje op het bureaublad.

2 – Hardware

Overzicht

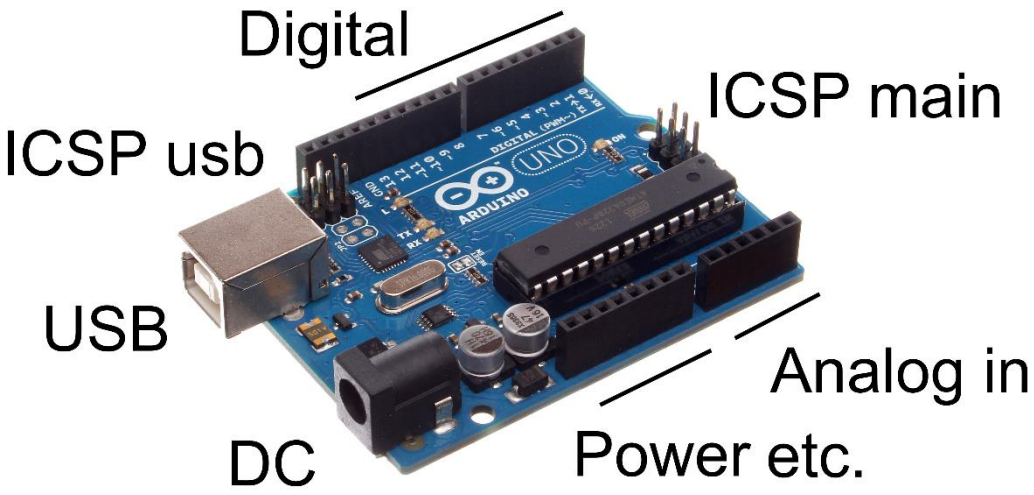
Een Arduino heeft vele mogelijkheden om te verbinden met de buitenwereld, maar ze zijn nogal duidelijk gegroepeerd.

In deze hele basiscursus is één soort voedingsspanning alles wat je nodig hebt: 5 V van de PC via de USB-kabel. Maar als je de Arduino apart wenst te gebruiken, kan hij in plaats daarvan via de DC-aansluiting worden gevoed. Frederiksen verkoopt een handige adapter waarmee je een gewone 9 V-accu kunt aansluiten op de DC-aansluiting, artikelnummer 663810.

Een rij digitale in-/output “pins” zijn aan te vinden aan één kant van de printplaat. Eigenlijk verschijnen ze als vrouwelijke connectoren waar kabels met mannelijke stekkers op passen – maar ze zijn meestal direct verbonden met individuele pins op de microcontrollerchip – dus hier zullen we ze “pins” blijven noemen.

Op de tegenoverliggende rand vind je een rij analoge inputs en een rij pins met verschillende spanningen – hier heb je meestal maar 0 V en 5 V nodig als voeding voor de circuits die je op de Arduino aansluit.

Er zijn ook twee groepen van 6 pins die elk op twee plaatsen uit het bord steken. Deze worden gebruikt voor ICSP – “In-Circuit System Programming” voor de hoofdprocessor en de USB-interface. Het gebruik van deze connectoren wordt in deze cursus niet behandeld.



Zorg voor je Arduino

Een Arduino is gewoon “een printplaat met een paar componenten”. Daaronder zitten allerlei componentpins en soldeerpunten die voordat je het weet kortsluiting maken als je de Arduino op een paperclip of een elektronisch onderdeel plaatst.

- Houd de tafel altijd vrij rond de Arduino om te voorkomen dat er iets onderdoor glipt.

Frederiksen Scientific verkoopt een standaard waarop de Arduino kan worden gemonteerd – stabiel en met ruimte onderin. Deze biedt ook ruimte voor een breadboard naast de Arduino – die we in veel van de volgende modules van deze cursus gaan gebruiken.

Deze standaard heeft artikelnr. 663820.

3 – De Arduino op nul zetten

Een programma dat niets doet

In de Arduino-gemeenschap wordt een programma om een of andere reden een “sketch” genoemd, en dit wordt bijvoorbeeld gebruikt in het menusysteem van de Arduino IDE. In deze cursus zullen we die term niet gebruiken.

Wanneer je een nieuw Arduino programma gaat schrijven, zal het onderstaande sjabloon al aanwezig zijn in het codevenster:

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

}

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

}
```

Hier in het begin gaan we niet diep in op elk woord en elk haakje – wees simpelweg voorzichtig als je nieuwe tekst moet invoeren. Gaandeweg zullen de details ook duidelijk worden!

De twee regels die beginnen met "//" zijn *opmerkingen* – tekst die niets betekent voor het programma, maar die voor de lezer uitlegt wat er aan de hand is. (We zullen hier nog vele voorbeelden van zien.)

Het is duidelijk dat het sjabloon in twee delen is gerangschikt:

- Een blok met de aanduiding “setup”: De hier geschreven programmacode wordt *eenmalig* voor de rest van het programma uitgevoerd.
- Een blok met de aanduiding “loop”: Deze code wordt keer op keer herhaald.

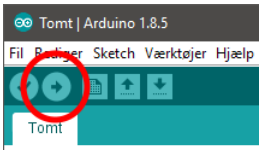
Elk programmablok is omgeven door **{** accolades **}**,

Het plan is om (zeer binnenkort) een heel eenvoudig testprogramma te maken om te kijken of de opstelling van de IDE, de kabelverbindingen etc. werken. Maar je Arduino heeft waarschijnlijk al een soortgelijk programma geüpload, dus we zullen het bovenstaande – volledig lege – programma gebruiken voor de “zeroing” van de Arduino.

Sla het programma op met de naam “Empty”: Het menu *File / Save as*. Voer de naam “Empty” in en klik op “Save”. Dit leidt tot twee dingen: Een map genaamd “Empty” wordt aangemaakt, en in deze map wordt het programma opgeslagen als “Empty.ino”. (Het is afhankelijk van jouw Windows of de extensie “ino” kan worden getoond, maar je kunt Arduino bestanden altijd herkennen aan het groene icoontje.)

Arduino programma's moeten altijd in een aparte map staan met dezelfde naam als het programma.

Het programma uploaden



Nu moet het programma geüpload worden naar de Arduino. Klik op de knop met de pijl:

Als het goed is zorgt dit ervoor dat er drie dingen gebeuren:

1. Het programma wordt vanuit de leesbare tekst vertaald naar een vorm die geschikt is voor de microcontroller. Dit wordt *compilatie* van de code genoemd en het deel van het systeem dat deze taak uitvoert wordt een *compiler* genoemd. Nu is het programma omgezet in *machinecode*.
2. Het programma (d.w.z. de machinecode) wordt via de USB-aansluiting naar de Arduino gestuurd.
3. De Arduino voert het programma uit.

Tijdens dit proces kun je de voortgang eerst op de PC volgen, daarna op de Arduino.

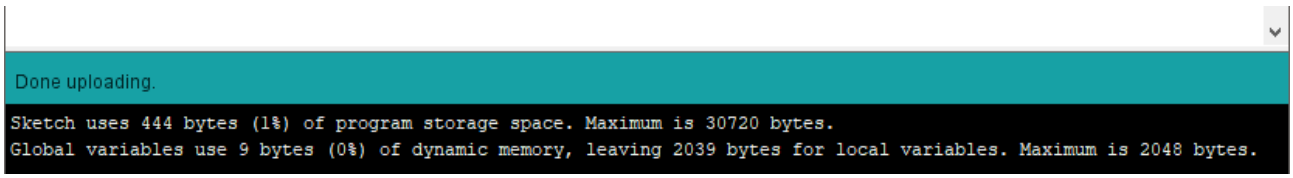
Wat je kunt zien op de PC

In de groene balk onder het lichtcodevenster verschijnen drie statusmeldingen na elkaar:

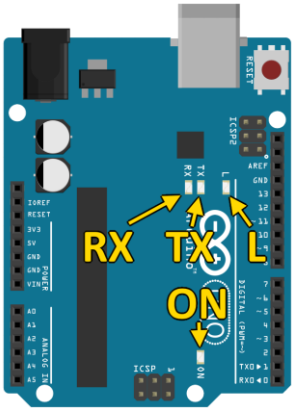
- Compiling sketch…
- Uploader…
- Done uploading

In de zwarte ruimte hieronder wordt een meer gedetailleerde status getoond. Als alles goed is, zal het geheugengebruik van het programma worden getoond. Je zult merken dat zelfs dit lege programma een beetje ruimte in het Arduino geheugen inneemt.

Uiteindelijk zal de onderkant van het programmavenster er zo uitzien:



Wat je kunt zien op de Arduino



Een Arduino Uno rev. 3 bevat vier kleine LED's – zie tekening:

- ON: Licht op wanneer de Arduino aanstaat
- RX: "Receive" – Licht op wanneer je informatie van de PC verstuurt via de USB-aansluiting
- TX: "Transmit" – Licht op wanneer informatie van Arduino naar de PC wordt gestuurd
- L : Een extra LED die vanuit het programma kan worden aangestuurd – zo meteen zul je zien hoe. (Knippert ook tijdens het uploaden.)

Als alles in orde is, zullen de *RX*, *TX* (en *L*) LED's knipperen terwijl het programma wordt geüpload. Daarna zal *L* constant branden.

Nu is de Arduino vrijgemaakt, en ben je klaar om je eerste "echte" programma te schrijven.

Als het niet werkt

- Is er een probleem met het programma? (Zeer onwaarschijnlijk met dit programma! – maar probeer ergens een "x" te schrijven en klik opnieuw op Upload. Kijk hoe de foutmelding eruit ziet. Verwijder de "x" weer ☺)
- Is de USB-kabel zowel op de Arduino als de PC aangesloten? (Probeer te zien hoe het systeem reageert als je probeert te uploaden met de kabel losgekoppeld.)
- Is het juiste board geselecteerd? (Menu *Tools / Board*. Normaal is dit "Arduino / Genuino Uno".)
- Is de juiste Port geselecteerd? (Trek de USB-kabel eruit. Begin bij het begin in het menu *Tools / Port*. Onthoud de lijst in het menu. Sluit de kabel weer aan. Begin bij het begin in het menu *Tools / Port*. Kijk naar de lijst. De nieuwe port op de lijst is de juiste.)

4 – Het eerste Arduino programma

Invoeren en testen van het programma

De IDE slaat het programma automatisch op onder zijn huidige naam telkens als het wordt gecompileerd (als er wijzigingen zijn aangebracht). Het is daarom een goede gewoonte om het programma altijd onder een *nieuwe* naam op te slaan, voordat er grotere wijzigingen worden aangebracht. Sla op met de naam "Blink-LED".

Voer het programma in, precies zoals hieronder weergegeven – waar de twee commentaarregels zijn weggelaten. Het is belangrijk om grote en kleine letters, haakjes en puntkomma's te gebruiken zoals afgebeeld.

```
void setup() {
  pinMode(13,OUTPUT);
}

void loop() {
  digitalWrite(13,HIGH);
  delay(250);
  digitalWrite(13,LOW);
  delay(750);
}
```

We zullen het programma zo meteen doornemen. Maar eerst gaan we proberen of het werkt! Klik nogmaals op de Upload-knop. Zodra het programma is overgezet naar de Arduino, zal LED *L* elke seconde een korte flits geven.

Als het niet werkt, moet je weer terug naar de checklist hierboven.

Het programma begrijpen
De Arduino heeft een reeks genummerde pins die gebruikt worden voor de communicatie naar de buitenwereld. Voordat een pin kan worden gebruikt, moet de Arduino weten of het een input of een output moet zijn. Arduino Uno heeft de LED *L* aangesloten op pin 13 die dus een output moet zijn, dit wordt gedaan met de regel:

```
pinMode(13,OUTPUT);
```

`pinMode` is een van de ingebouwde functies in het Arduino systeem. De functie wordt aangeroepen met twee *parameters*: het pinnummer en een specificatie van de functie. `OUTPUT` is een constante, gedefinieerd door het systeem. (Deze heeft een waarde van 1, maar het is veel gemakkelijker om het programma te begrijpen bij het gebruik van constanten met herkenbare namen.)

Zodra de pin als output is gedefinieerd, blijft dit zo – daarom wordt deze instructie in het `setup`-blok geplaatst.

Op dezelfde manier is `digitalWrite` een voorgedefinieerde functie met twee parameters. De eerste parameter is het pinnummer, de tweede parameter is de gewenste toestand. De constanten `HIGH` en `LOW` worden gebruikt voor het instellen van de output logical *high* (ca. 5 V op een Arduino Uno), resp. logical *low* (ca. 0 V). De LED brandt als de output hoog is.

Tot slot is `delay` een functie met één parameter. De functies doen niets gedurende een door de parameter gegeven periode. De eenheid is milliseconden, waardoor 250 overeenkomt met een kwart van een seconde. Als dit tijdsvenster voorbij is gaat het programma verder met de volgende instructie.

De regels in het `loop`-blok zullen samen de LED een kwart seconde aanzetten en vervolgens driekwart seconde uitdoen – wat tot in het oneindige wordt herhaald.

Uitdaging 1
Sla het programma op met een nieuwe naam: “BlinkTwice”.

Herschrijf het programma om elke seconde twee korte flitsen en een langere pauze te geven.

Open het originele “Blink-LED” programma opnieuw. Zoals je ziet zijn beide programma's – elk in een eigen venster – tegelijkertijd gemakkelijk toegankelijk. Probeer de twee programma's afwisselend te uploaden.

5 – Leesbaarheid van programma's
Je doet jezelf een groot plezier door het programma netjes op te maken met regelopschuiving en inspringen – zoals in het voorbeeld hierboven. Dit geldt vooral wanneer de code langer wordt dan 20-30 regels.

De compiler die de programmatekst vertaalt naar machinetekst geeft absoluut niets om zijn uiterlijk. Hij controleert alleen of elke instructie correct is en stopt met een puntkomma. Het programma zou gemakkelijk zo geschreven kunnen worden:

```
void setup(){pinMode(13,1);}void loop(
){digitalWrite(13,1);delay(250);digitalWrite(13
,
0);delay(750
);}
```

- maar de meeste mensen zouden beamen dat de originele versie een stuk makkelijker te lezen en te begrijpen is.

6 – Schrijfstijl
In de rest van deze cursus zullen we bijvoorbeeld “`pinMode()`” in de tekst schrijven als we het over “de functie pinMode” hebben. Het haakje is er om je eraan te herinneren dat we te maken hebben met een functie en niet met bijv. een constante.

In programmalijsten worden de haakjes uitgebreid, inclusief parameters, zoals in “`pinMode(13,OUTPUT)`”.

7 – Help!
Selecteer het menu-item *Help / Reference* voor een handleiding van de Arduino programmeertaal en de ingebouwde functies. Het is een goede plek om te zoeken naar meer dan wat in deze basiscursus wordt behandeld.

Zoals alle naslagwerken kan de handleiding in het begin wat overweldigend zijn – je leert er geleidelijk aan steeds beter mee om te gaan.