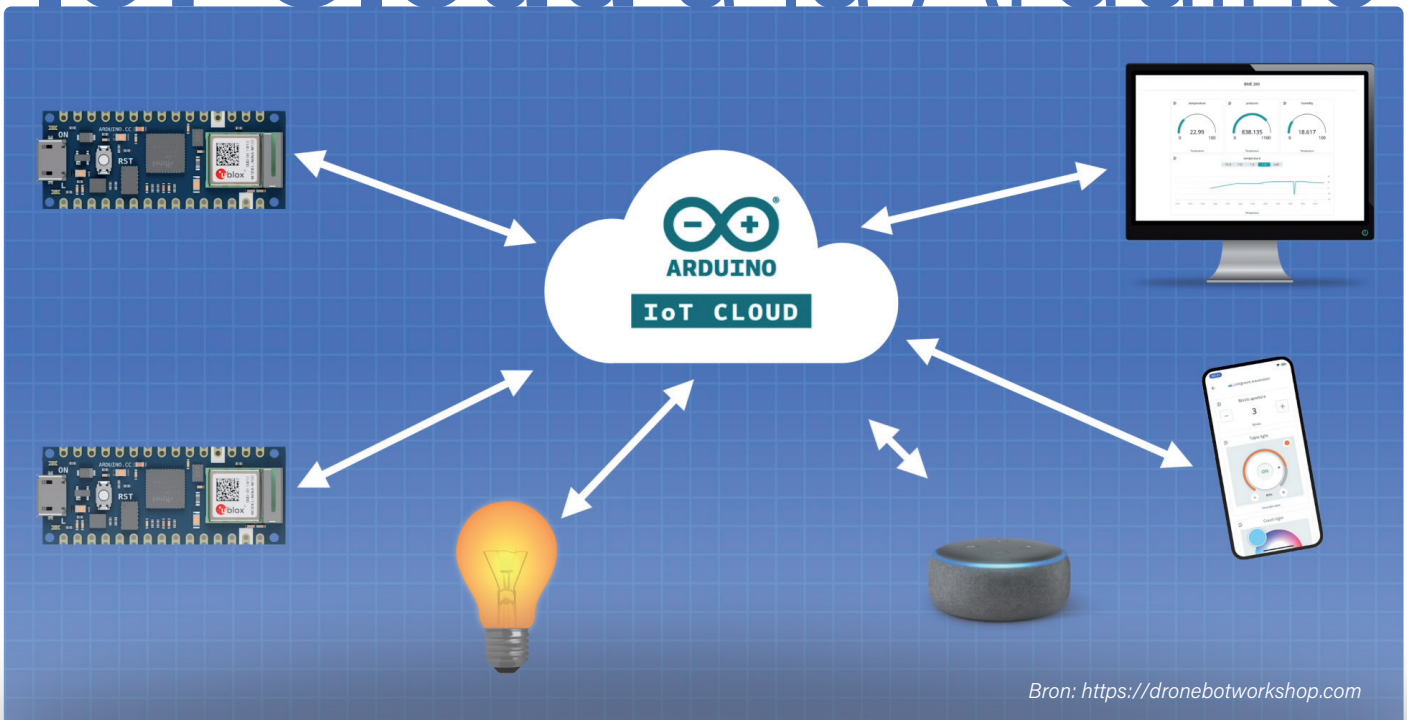
IoT Cloud à la Arduino



Bron: https://dronebotworkshop.com

Tam Hanna (Slowakije)

De Arduino IoT Cloud biedt ontwikkelaars van IoT-toepassingen een eenvoudige oplossing voor het implementeren van een cloud back-end zonder met MQTT te hoeven stoeien. Nieuwsgierig? Lees dan verder!

Veel microcontrollertoepassingen hebben tegenwoordig Internet of Things-functionaliteit (IoT), waarmee informatie wordt gedeeld via IoT-cloudservices en een MQTT-broker. Het maken van zo'n toepassing met een lokale ontwikkelomgeving zoals de traditionele Arduino-IDE kan een lastige klus zijn. Bij de Arduino Cloud bevindt de IDE zich in de cloud, dus uw browser wordt een venster op de IDE. We hebben het uitgeprobeerd door de waarde van een variabele naar de cloud te sturen en daarmee een LED te doen oplichten. Daarna hebben we geprobeerd de werking te verstoren. De basis van alle IoT-apparaten is natuurlijk een *Thing*. In de Arduino IoT Cloud-ontwikkelomgeving is een *Thing* een virtueel object in de cloud. In de echte wereld is het een object zoals een server, een controllerkaart of een soortgelijk 'intelligent' apparaat [1]. Uw *Thing* wordt in de cloud opgebouwd door met een online-editor een sketch te schrijven die bepaalt hoe het zich moet gedragen aan de hand van een aantal *Variables*.

Voor wie is de Arduino IoT Cloud bedoeld?

Het is belangrijk om vooraf te vermelden dat de Arduino IoT Cloud geen alternatief is voor andere cloud computing-platforms, zoals Amazone AWS IoT Core, Microsoft IoT Hub of Yandex IoT Core. Als u veel apparaten en data wilt beheren, zijn die gevestigde services de beste oplossing.

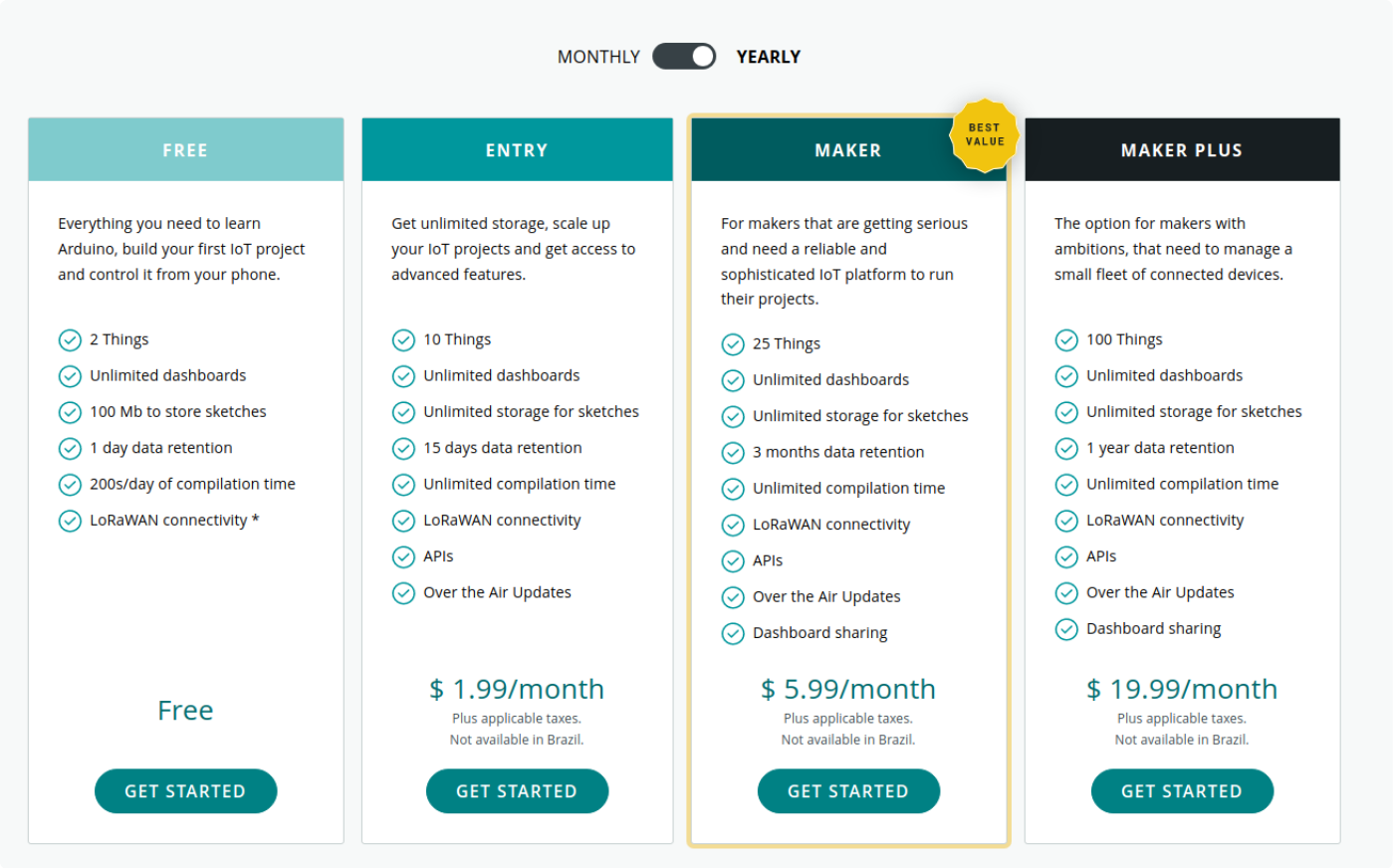
Bij de introductie van de nieuwste versie van de Arduino IoT Cloud omschreef Massimo Banzi, CTO bij Arduino, zijn ambities voor het platform als volgt: "Arduino biedt nu een compleet platform met de MKR-familie; een gestroomlijnde manier om lokale IoT-knooppunten en -randapparaten te creëren. Deze gebruiken diverse verbindingsmogelijkheden en zijn compatibel met hardware, gateways en cloud-systemen van derden. Met de Arduino IoT Cloud kunnen gebruikers niet alleen Arduino-hardware maar ook de meeste Linux-apparaten beheren, configureren en verbinden; een echte democratisering van de IoT-ontwikkeling."

De ondersteuning van de Arduino MKR-boards voor het IoT en van enkele populaire boards van derden is een nuttige toevoeging en kan voor veel IoT-ontwikkelaars aanleiding zijn om dit toegankelijke ontwikkelplatform nog eens te bekijken.

Opzetten van de hardware

Alleen de meest basale van de vier beschikbare versies van het Arduino Cloud-platform kan gratis worden gebruikt. De verschillende abonnementen en hun mogelijkheden zijn weergegeven in **figuur 1**. De ondersteuning van de Arduino Cloud voor kaarten van derden, zoals de populaire ESP8266- en ESP32-familie, en de lijst van compatibele platforms [2] zijn interessant voor de hele maker-community (**tabel 1**). Met behulp van driver-bibliotheken kunnen ook verschillende Linux-systemen informatie wegschrijven naar en ophalen uit de Arduino-cloud.

We gebruiken hier een Arduino Nano RP2040 Connect-module als



Figuur 1. Welk plan past het best bij u? (Stand: 20 januari2022.)

doelsysteem om de Arduino Cloud uit te proberen. Deze kaart is opgebouwd rond de RP2040-microcontroller van de Raspberry Pi Foundation en bevat een u-blox WiFi-module. Voordat we de Arduino Cloud gaan configureren, sluiten we de kaart via een USB-kabel aan op een computer.

We moeten nu eerst naar de website [3] om een nieuw Arduino-account aan te maken. We beperken ons even tot het gratis basisplan, dat bedoeld is voor nieuwkomers. Dan klikken we op de knop *Create Thing* om een nieuw 'ding' aan te maken.

Het overzicht in **figuur 2** toont de configuratie van de drie basiscomponenten van de ontwikkelomgeving. Ik heb een eerder Arduino-account gewist, zodat ik met een schone lei kon beginnen. De volgende stappen zijn uitgevoerd op een computer met Windows 10, maar het proces is hetzelfde op een Ubuntu Linux-systeem. De hardwaredetectie werkt meestal beter onder Unix.

Nu kunnen we op de snelkoppeling onder *Device* klikken; dan kiezen we voor *Set Up an Arduino Device*. Een paar seconden later geeft het back-end aan dat een component met de naam *Arduino Create Agent* ontbreekt. Klik op de knop *Download* om de software te downloaden en installeer die op de normale manier.

Let op: *Create Agent* is browser-specifiek: als u installeert met Chrome, moet u opnieuw installeren. Als er firewall-waarschuwingen verschijnen, moet u die bevestigen. Sta zowel toegang voor privé- als openbare netwerken toe. Nu verschijnt de *Arduino Create Agent* in de taakbalk. Soms moet die eerst worden opgeroepen vanuit het startmenu, maar in elk geval is de driver-installatie nu afgerond.

We verversen nu de weergave in de browser tot de Arduino Cloud meldt dat onze Nano RP2040 Connect-kaart is herkend. Klik nu op

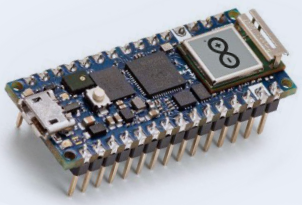
Tabel 1. De Arduino Cloud ondersteunt deze boards.

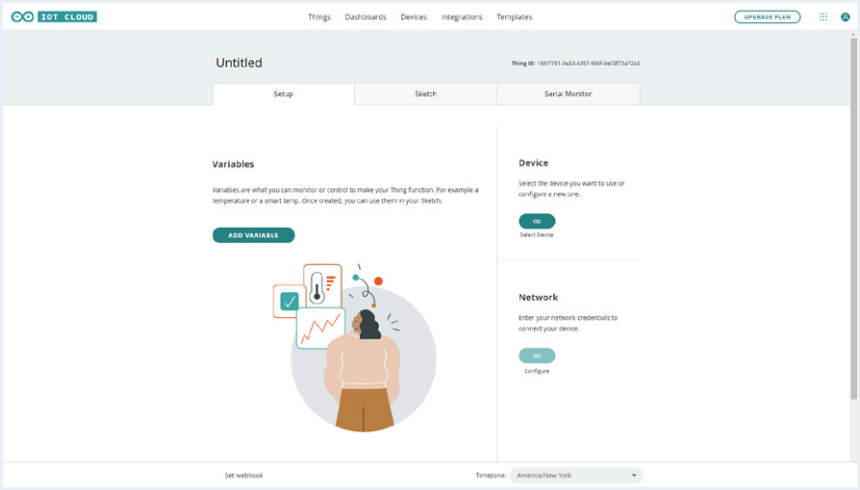
WLAN
- MKR 1000 WiFi
- MKR WiFi 1010
- Nano RP2040 Connect
- Nano 33 IoT
- Portenta H7

LoRaWAN
- MKR WAN 1300
- MKR WAN 1310

GSM/NB-IoT
- MKR GSM 1400
- MKR NB 1500

ESP32/ESP8266
- een groot aantal boards van derden





Figuur 2. De IoT Cloud leidt de ontwikkelaar stap voor stap naar het doel.

Figuur 3. Deze Arduino is verbonden met de Cloud.

de knop *Configure* om de configuratie-wizard te starten. Die vraagt u om een 'gebruikersvriendelijke' naam in te stellen en initialiseert dan het doelsysteem met de basis-communicatiesoftware. Onder Windows kunnen er netwerkproblemen optreden. Onder Linux werkt alles een stuk betrouwbaarder. Soms wordt het *Network*-gedeelte niet automatisch ingeschakeld door de Arduino Cloud. Het is alleen beschikbaar als u één van de variabelen voor datacommunicatie aanmaakt.

Klik nu op *Variables* om het dialoogvenster voor het toevoegen van een variabele te openen. We geven hem de naam `ledIntenBool` en het datatype *Boolean*. U zult zien dat de Arduino Cloud niet alleen de bekende C-datatypen ondersteunt, maar ook 'wrappers' bevat voor allerlei grootheden uit de echte wereld.

Als u zich wilt beperken tot C-typen, kunt u het beste de optie *Basic Types* aanvinken. Het is in theorie mogelijk om dan instellingen onder *Variable Permission* en *Variable Update Policy* aan te passen, maar de standaardinstellingen zijn goed genoeg voor ons doel, daarom sluiten we het dialoogvenster. Vervolgens maken we een veld van het type *Integer Number*, dat we de naam `ledIntenInt` geven.

De code

Na het aanmaken van de variabelen zien we rode stippen in de tab *Sketch*, en dat wijst op een verandering in de programmastructuur. We kunnen nu het *Network*-gedeelte op de snelkoppeling voor de WiFi-instellingen klikken. Ik vind het het prettigste om een commandoregel-tool zoals *iwlist* te gebruiken op een Linux-machine en de gegevens naar het klembord te kopiëren.

Daarna gaan we naar de tab *Sketch* en klikken we op de knop *Verify and Upload*. De Arduino Cloud gaat de code dan compileren en stuurt die naar de aangesloten RP2040 met behulp van de *Arduino Cloud Agent*. Als de gecompileerde code succesvol is afgeleverd, verschijnt de melding "*Untitled_dec25a uploaded successfully on board Arduino Nano RP2040 Connect (/dev/ttyACM0)*".

Na de verplichte reset begint de RP2040 naar huis te bellen via de WiFi-interface. Na een poosje en nadat we diverse malen op F5 hebben gedrukt, zien we de melding "*Status: online*" zoals in **figuur 3**.

Als u hebt gekozen voor één van de uitgebreide Arduino Cloud-accounts, kunt u rechtstreeks software-updates ontvangen via WiFi; voor onze eenvoudige experimenten met een gratis account is een bedrade verbinding nodig.

In detail

De eenvoudige editor in de tab Sketch biedt weinig mogelijkheden, maar via de knop *Open full editor* krijgen we een complete cloud-gebaseerde IDE waarmee we gemakkelijk kleine projecten kunnen bewerken. Laten we eerst eens naar de inhoud van het bestand *thingProperties.h* kijken. Dit bevat de definities voor onze sketch. Om te beginnen zien we daar de volgende declaraties voor het WiFi-netwerk:

```
const char SSID[] = SECRET_SSID; // Network SSID (name)
const char PASS[] = SECRET_PASS; // Network password
// (use for WPA, or use as key for WEP)
```

De Arduino Cloud zorgt voor de invulling van de naam en het wachtwoord, die we eerder onder Network hebben ingevoerd. Dan worden twee variabelen gedeclareerd met de namen die we hebben opgegeven onder *Variables*

```
int ledIntenInt;
bool ledIntenBool;
```

De Arduino Cloud implementeert de variabelen 'in zijn "back-end' als standaard C-variabelen met wat extra eigenschappen. Deze eigenschappen zijn te vinden in de methode `initProperties`, die zorgt voor het opzetten van de primitieven en structuren die nodig zijn voor de communicatie met de cloud:

```
void initProperties() {
ArduinoCloud.setThingId(THING_ID);
ArduinoCloud.addProperty(ledIntenInt, READWRITE,
ON_CHANGE, onLedIntenIntChange);
ArduinoCloud.addProperty(ledIntenBool, READWRITE,
  ON_CHANGE, onLedIntenBoolChange);
}
```

Hier valt op dat de methode `addProperty` het 'registeren' van de attributen voor zijn rekening neemt. Hier worden pointers naar de

functies `onLedIntenIntChange` en `onLedIntenBoolChange` doorgegeven, die straks een belangrijke rol zullen spelen.
De besturingsfunctie van de applicatie wordt beschreven in de sketch, die begint met het opnemen van de header (hier niet getoond). Dan volgt de initialisatie van de sketch, en dat ziet er als volgt uit:

```
void setup() {
 Serial.begin(9600);
 delay(1500);

 initProperties();

 ArduinoCloud.begin(ArduinoIoTPreferredConnection);
 setDebugMessageLevel(2);
 ArduinoCloud.printDebugInfo();
}
```

Voor de Arduino-programmeeromgeving is de Arduino Cloud een gewone hardware-driver. Het globale object `ArduinoCloud` bevat een aantal functies die uw code gebruikt voor de communicatie met de driver in de cloud. Let op de aanroep van `setDebugMessageLevel`. Daarmee wordt de 'spraakzaamheid' van de driver ingesteld: hoe hoger de waarde, des te meer debuginformatie de clouddriver weergeeft via de seriële poort van het board.
Bij 'ingewikkelde' drivers is het altijd de vraag hoe het rekenvermogen verdeeld wordt. Het projectgeraamte dat de Arduino Cloud voor ons heeft gemaakt, regelt dat in de methode `loop()`, die het rekenvermogen als volgt toewijst:

```
void loop() {
 ArduinoCloud.update();
}
```

De Arduino Cloud geeft ons standaard de volgende drie listener-methodes:

```
void onTestScheduleChange() {
}
void onLedIntenBoolChange() {
}
void onLedIntenIntChange() {
}
```

`onLedIntenBoolChange` en `onLedIntenIntChange` zijn verantwoordelijk voor de variabelen in de cloud, terwijl `onTestScheduleChange` helpt bij het implementeren van 'interne' functies van de Arduino Cloud.
Bij de volgende stap regelen we dat door gebruik te maken van de ingebouwde 'intelligente' mogelijkheden van de Cloud. Het Arduino-board dat we hier gebruiken, is standaard voorzien van een (rode) LED aan pen 13 en een RGB-LED (kanalen LEDR, LEDG en LEDB), die via drie PWM-signalen in alle kleuren kan worden aangestuurd.

We gaan nu terug naar de functie `setup()` en initialiseren de nodige pinnen:

```
void setup() {
...
 ArduinoCloud.printDebugInfo();

 pinMode(LED_BUILTIN, OUTPUT);
 pinMode(LEDB, OUTPUT);
}
```

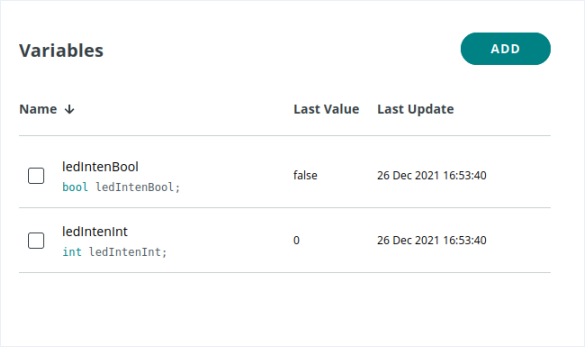
Veranderingen in de Cloud activeren de Listener, die de binnenkomende waarden naar de hardware schrijft:

```
void onLedIntenBoolChange() {
 digitalWrite(LED_BUILTIN, ledIntenBool);
}
void onLedIntenIntChange() {
 analogWrite(LEDB, ledIntenInt);
}
```

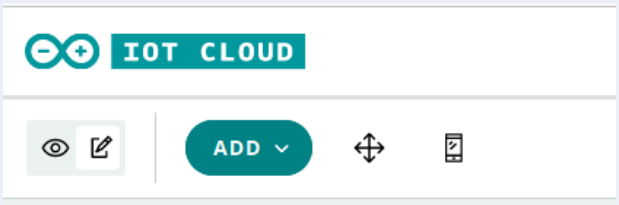
U kunt nu de sketch opnieuw naar de kaart sturen. De RGB-LED heeft een gemeenschappelijke anode, dus de individuele kleuren worden aangestuurd door een 0 naar de desbetreffende LED te schrijven. De variabelen worden geïnitialiseerd zoals weergegeven in **figuur 4**, zodat de blauwe diode van de RGB-LED oplicht na succesvolle initialisatie.

De inhoud van de variabelen veranderen

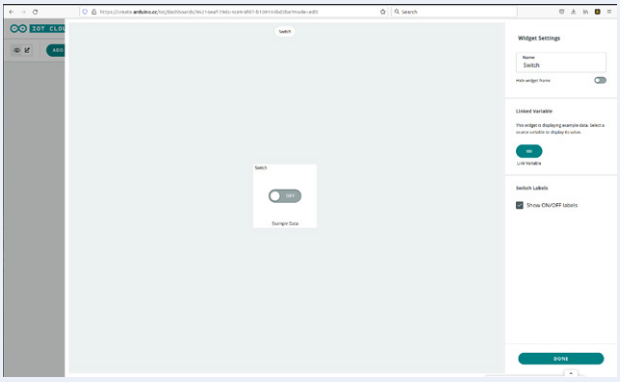
We gaan nu terug naar de Arduino Cloud en klikken op de tab *Dashboards*. Als u een nieuw account hebt, wordt u nu gevraagd om een eerste dashboard aan te maken. Klik op de knop *Build Dashboard* om de editor te starten. Dat kan even duren, zelfs als u een snelle Internetverbinding hebt.
Om elementen aan te passen en toe te voegen aan het dashboard klikt u op het edit-pictogram (potlood) links bovenaan het venster



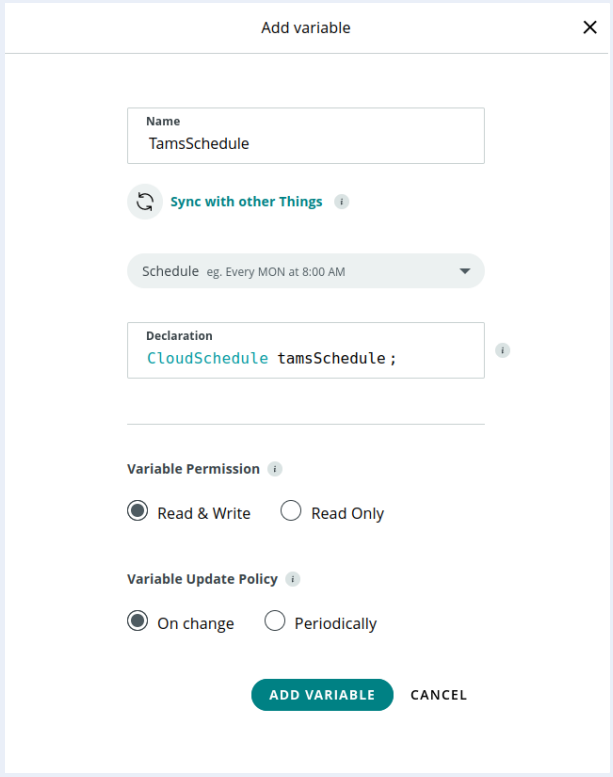
Figuur 4. De Arduino IoT Cloud geeft gedetailleerde informatie over de huidige toestand van de informatie in een variabele.



Figuur 5. Schakel het dashboard naar edit-modus door linksboven op het potlood-pictogram te klikken.



Figuur 6. De bewerkingsinterface voor besturingselementen is 'modaal.'



Figuur 7. De Cloud Scheduler is gewoon een datatype als alle andere.

(**figuur 5**). Als u eenmaal in de edit-modus bent, verschijnt de blauwe *ADD*-knop. Klik erop om te zien welke widgets er beschikbaar zijn. We kiezen eerst een *Switch*-widget, dat verschijnt zoals in **figuur 6**.
Rechts op het scherm ziet u nu het veld *Linked Variable* met een link-knop. Klik die aan voor een lijst van alle *Things* en *Variables* in uw account. Hier kiezen we de variabele `ledIntenBool` en die linken met de knop *Link Variable*. De toestand van `ledIntenBool` wordt nu bestuurd door de toestand van de Switch. Klik op DONE om de edit-mode te sluiten. De switch is nu deel van het dashboard. We kunnen nu op het oog-pictogram klikken om de schakelaar te activeren. De schakelaar op het dashboard bestuurt nu de rode LED naast de microUSB-connector.
Nu willen we ook de helderheid van de blauwe LED regelen. Zet het dashboard weer in de edit-modus en voeg een nieuw besturingselement toe met *Add -> Widgets*. Dit keer gebruiken we een *Slider*. We stellen de *Value Range* in op 0 - 255. De link wordt gemaakt met de variabele `ledIntenInt`, die staat voor de helderheid van de RGB-LED. Tenslotte schakelen we naar de activeringsmodus en zien dat de stand van de schuifregelaar nu de helderheid van de blauwe LED bepaalt.

Gebruik van de Scheduler

Op het moment van schrijven is er een nieuwe functie voor het uitvoeren van ingeroosterde taken of web cron-jobs: deze maakt gebruik van een variabele met de naam *CloudSchedule*, die u kunt configureren om op bepaalde tijdstippen en gedurende een bepaalde periode true of false te zijn. U hoeft geen timer-functie te gebruiken want deze variabele wordt door de Arduino IoT Cloud automatisch geset of gereset, afhankelijk van uw instelling. Nu kunnen taken worden getriggerd door de waarde van deze variabele te controleren. Om dat te demonstreren, maken we een nieuwe variabele van het type *Schedule*. **Figuur 7** toont hoe dat er uitziet. We kunnen dit nieuwe type nu ook zien in de code:

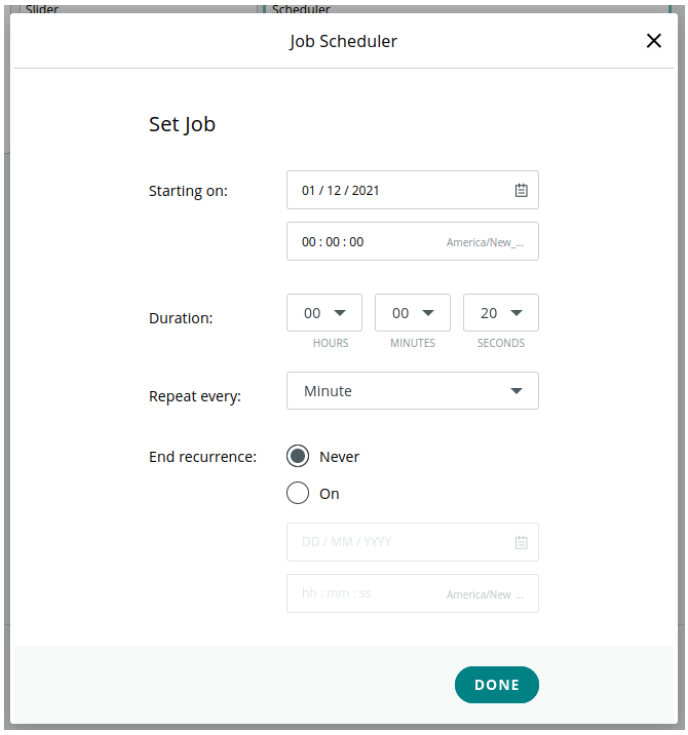
```
CloudSchedule tamsSchedule;
```

De tijdparameters voor deze variabele worden geconfigureerd via het dashboard. In **figuur 8** ziet u de besturingselementen waarmee u dat kunt doen.
Nu kunnen we de 'lokale' verwerking van de waarden in `tamsSchedule` gaan programmeren:

```
void setup() {
...

 pinMode(LED_BUILTIN, OUTPUT);
 pinMode(LEDB, OUTPUT);
 pinMode(LEDR, OUTPUT);
}

void loop() {
 ArduinoCloud.update();
 // Your code here
```



Figuur 8. De Scheduler wordt met het Dashboard geconfigureerd.

```
 if(tamsSchedule.isActive()){
  digitalWrite(LEDR, HIGH);
 }
 else{
  digitalWrite(LEDR, LOW);
 }
}
```

De ontwikkelaar moet dus helemaal zelf zorgen voor de verwerking van de informatie in `tamsSchedule`. De Cloud doet niets anders dan de waarde in `tamsSchedule` periodiek actualiseren. De continue polling-procedure die we hier in `loop()` toepassen, is vast niet de optimale manier maar het werkt prima. We kunnen het programma nu naar de Arduino sturen en dan kunt u het periodiek rood oplichten van de RGB-LED bekijken.
Toen ik zo ver gekomen was, kon ik het niet laten om uit te proberen wat er zou gebeuren als de internetverbinding wegvalt. Ik zette mijn WiFi uit en de Arduino begon ongecontroleerd het blauwe element van de RGB-LED en ook de LED op pen 13 aan te sturen. Na een paar seconden startte hij helemaal opnieuw op.
Op het moment dat dit artikel naar de drukker ging, was het me nog niet helemaal duidelijk hoe de Arduino Cloud zich herstelt na het wegvallen van de radioverbinding tussen het eindapparaat en de server.

Een handige mogelijkheid

Het is duidelijk dat er nog gewerkt wordt aan de Arduino IoT Cloud. Hij wordt constant verder ontwikkeld en verbeterd. Het platform biedt heel fraaie mogelijkheden en geeft de ontwikkelaar van IoT-toepassingen een laagdrempelige mogelijkheid voor het implementeren van een cloudserver zonder met MQTT en dergelijke te hoeven worstelen. Ondanks wat kleine probleempjes kan ik dit product van harte aanbevelen!

210550-03

Een bijdrage van

Idee, illustraties en tekst: **Tam Hanna**
Redactie: **Rolf Gerstendorf**
Vertaling: **Evelien Snel**
Layout: **Harmen Heida**

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de auteur via tamhan@tamoggemon.com of naar de redactie van Elektor redactie@elektor.com.

WEBLINKS

[1] Digital twin: https://en.wikipedia.org/wiki/Digital_twin
[2] De Arduino Cloud ondersteunt deze boards: https://bit.ly/3t8Vl3W
[3] Arduino Things: https://create.arduino.cc/iot/things