CE

2

Alle Versuche im Überblick

Internet of Things calendrier de l'Avent 2017

Lorsqu'il s'agit de Cisco, il y aura plus de 50 milliards de périphériques en réseau en 2020, Intel est encore plus optimiste et annonce 200 milliards de périphériques1. Chacun de ces périphériques – appelés également « Objet » – possèdent une adresse unique et communique avec le monde extérieur via internet ou via une autre interface, telle que Bluetooth. De la machine à café au réfrigérateur, de la voiture au train, de la machine de production jusqu'au bracelet, tout est programmable et peut communiquer avec d'autres objets. Ce sujet est plus que du battage médiatique, il s'agit de la pointe de la technique et par conséquent nous devrions tous nous en préoccuper. Profitez donc de la période de l'Avent et grimpez dans l'Internet des objets , ou Internet of Things (IoT). En 24 expériences vous appréhenderez le sujet et programmerez votre propre objet. Nous vous souhaitons beaucoup de plaisir !

Code source et informations supplémentaires

Dans les 24 prochains jours vous recevrez beaucoup de nouvelles informations sur le sujet IoT (Internet of Things) et réaliserez des projets passionnants. Pour que vous ne soyez pas en partie obligé de taper des programmes lourds, le code source et si nécessaire d'autres informations sont prêtes au téléchargement. Pour cela allez sur la page **http://www.buch.cd** et saisissez le code **15007-3**. Là se trouve une archive à télécharger. Dans l'archive se trouve un répertoire pour chaque jour. Lisez également le fichier **Lisez-moi.pdf** dans l'archive.

À partir du jour 8 les programmes sont partiellement très lourds, par conséquent ils ne sont pas complètement reproduits. Seuls les éléments du projet sont décrits dans le manuel, dont vous avez absolument besoin pour la compréhension et la mise en œuvre. Pour les projets qui sont contrôlés par une appli Smartphone spécialement conçue, le téléchargement complet est disponible pour chacun d'eux. Si vous voulez procéder à des modifications sur le programme, regardez toujours d'abord le programme fini, faites une copie et modifiez ensuite la copie. Si quelque chose doit mal se passer, vous trouverez les fichiers source sur **http://www.buch.cd**.

Mises à jour dans l'espace de téléchargement
Le calendrier de l'avent a été conçu bien avant la période de l'avent et se base sur les versions de programme de cette période. S'il devait y avoir d'importantes modifications avant l'Avent, vous trouverez dans l'espace de téléchargement une mise à jour des projets concernés.

Connaissances de base des composants

Un composant est inclus dans chaque discipline. À ce stade, vous allez apprendre des informations plus importantes concernant les composants.

DELs

Les DELs (diodes lumineuses) s'allument lorsque le courant traverse dans le sens du flux. Les DELs dans les circuits sont illustrées par un symbole triangulaire en forme de flèche qui indique le sens du flux de la borne positive au pôle négatif ou au fil de terre. Une DEL laisse passer presque autant de courant dans la direction du flux et n'a qu'une très faible résistance. Pour limiter le courant d'écoulement et ainsi empêcher que la DEL brûle, une résistance en série de 220 ohm doit généralement être intégrée entre la borne de raccordement et l'anode de la DEL ou entre la cathode et borne de terre. Cette résistance en série protège également la sortie de la carte IoT d'intensités de courant trop élevées. Les DELs dans le calendrier de l'Avent ont une résistance en série déjà intégrée et peuvent être branchées directement sur la borne de la carte IoT.

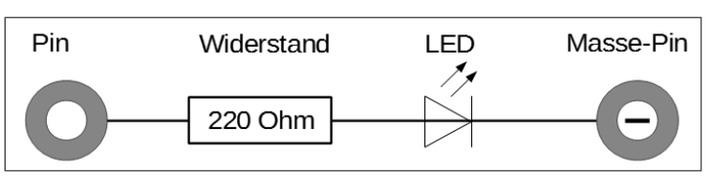


Schéma électrique d'une DEL avec résistance en série

1 Source : https://www.fool.com/investing/general/2016/01/18/internet-of-things-in-2016-6-stats-everyone-should.aspx

Dans quel sens brancher la DEL ?
Les deux fils de raccordements d'une DEL ont une longueur différente. Le plus long est le plus, l'anode, le plus court, la cathode. Simple à mémoriser : le signe plus a un trait de plus que le signe moins et rend ainsi le fil plus long. D'autre part, la plupart des DELs sont aplaties sur le côté moins, comme un signe moins. Simple à mémoriser : cathode = court = bord.

Résistances et leurs codes couleurs

Les résistances sont utilisées pour limiter le courant sur des composants électroniques sensibles et comme résistances en série pour les DELs. L'unité de mesure pour les résistances est l'ohm. 1 000 ohms correspondent à 1 kilo ohm, abrégé k ohm. 1 000 k ohms correspondent à 1 méga ohm, abrégé M ohm. Souvent on utilise le signe oméga Ω pour l'unité ohm.

Couleur	Valeur de résistance en ohm			
	1. Anneau (dizaines)	2. Anneau (unités)	3. Anneau (multiplicateur)	4. Anneau (tolérance)
Argent			10^{-2} = 0,01	±10 %
Or			10^{-1} = 0,1	±5 %
Noir		0	10^{0} = 1	
Marron	1	1	10^{1} = 10	±1 %
Rouge	2	2	10^{2} = 100	±2 %
Orange	3	3	10^{3} = 1.000	
Jaune	4	4	10^{4} = 10.000	
Vert	5	5	10^{5} = 100.000	±0,5 %
Bleu	6	6	10^{6} = 1.000.000	±0,25 %
Violet	7	7	10^{7} = 10.000.000	±0,1 %
Gris	8	8	10^{8} = 100.000.000	±0,05 %
Blanc	9	9	10^{9} = 1.000.000.000	

Les anneaux de couleurs sur les résistances fournissent la valeur de la résistance. Celles-ci sont nettement plus faciles à reconnaître que de minuscules chiffres que l'on trouve encore seulement sur des résistances anciennes, et avec un peu de pratique vous pouvez facilement « traduire » les valeurs à partir des codes de couleurs.

La plupart des résistances ont 4 anneaux de couleurs. Les deux premiers anneaux de couleurs remplacent les chiffres, le troisième désigne un multiplicateur et le quatrième la tolérance. Cet anneau de tolérance est souvent de couleur or ou argent, couleurs qui n'existent pas sur les deux premiers anneaux. Ainsi, l'ordre de lecture est clair. La valeur de tolérance joue pratiquement aucun rôle dans l'électronique numérique. La carte indique la signification des anneaux de couleurs sur les résistances.

Le sens dans lequel la résistance est intégrée n'a pas d'importance. Par contre pour les DELs le sens de montage a un rôle important.

1er jour

Aujourd'hui dans le calendrier de l'avent

· 1 IoT Bluething Board2

Aujourd'hui vous apprenez à connaître la carte avec laquelle vous réaliserez les projets pendant les 24 prochains jours. En préparation pour les prochains jours, vous installerez le pilote pour la connexion USB, l'Arduino IDE, et à la fin, vous créerez un programme initial pour la carte.

Chipset sur la carte IoT
La carte IoT est fournie avec deux chipsets. Il y a un ATmega328P sur la carte pour l'exécution des programmes. Ce micro contrôleur communique avec un HC-05 via l'interface série. Le HC-05 est chargé de la liaison radio (Bluetooth). Le module supporte le dispositif Bluetooth V2.0+EDR. Il faut un Smartphone Android pour la communication de l'appli.

Configurer la carte IoT

Pour mettre en service la carte IoT, vous avez besoin d'un PC sous Linux, Mac OS X ou Windows et d'un câble micro USB. Ce câble de raccordement sert à l'alimentation en courant et à la connexion de la carte-mère IoT avec le PC pour programmer cela. Vous n'avez pas besoin d'acheter un tel câble, car vraisemblablement vous en avez un ; presque tous les Smartphones modernes utilisent ce type de fiche.

Sélectionner le port USB correct sur le PC
Si possible, branchez le câble à un port USB 2.0 de votre PC. Avec des ports USB 3.0 il peut y avoir des problèmes de connexion. Un port USB 3.0 se reconnaît souvent à une prise femelle bleue.

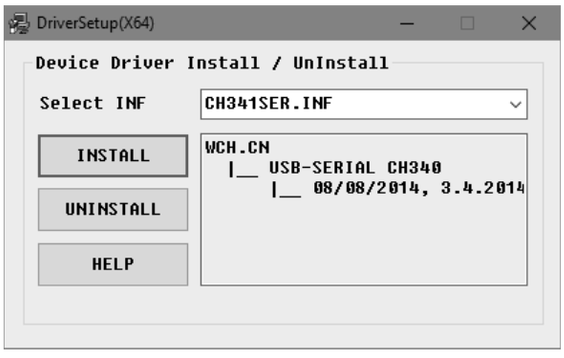
Les étapes suivantes sont nécessaires avant de connecter la carte au PC :

· Étape 1 : Installation du driver pour la carte IoT

· Étape 2 : Installation de l'Arduino IDE

Étape 1 : Installation du driver pour la carte IoT

Le port USB qui se trouve sur la carte IoT est connecté avec un chipset CH340G. Pour utiliser ce chipset pour la connexion USB, vous devez installer le driver adapté à votre système d'exploitation. Pour cela, exécutez les quatre étapes suivantes :



Installation du pilote du périphérique

1 Téléchargez le programme exemple et le pilote du périphérique sur **http://www.buch.cd**. Saisissez ici le code **15007-3** et suivez les indications sur l'écran.

2 Décompressez l'archive zip dans un dossier quelconque à l'intérieur de votre répertoire personnel.

3 Branchez votre carte IoT par le câble USB et lancez ensuite l'installation du driver avec le fichier `CH341SER.EXE`. Pour l'installation vous devez éventuellement confirmer une requête du contrôle de compte utilisateur.

4 Cliquez dans la boîte de dialogue de l'installation sur **Install** et attendez la confirmation que le driver a été installé.

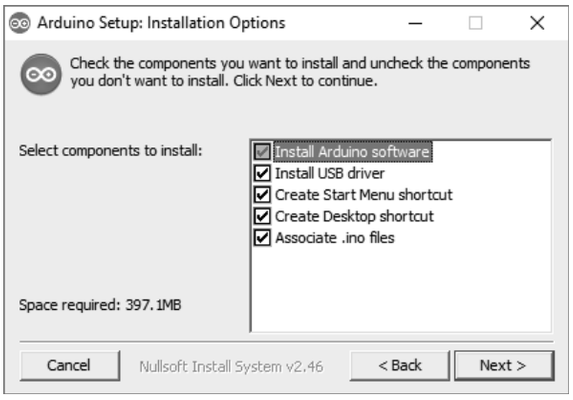
Étape 2 : Installation de l'Arduino IDE

La carte IoT est compatible avec Arduino Nano et peut être programmée avec l'Arduino IDE3. Écrivez les programmes dans l'Arduino IDE en langage C et transférez ensuite directement sur la carte IoT. Après le transfert le programme tourne sans connexion sur le PC, par conséquent vous pouvez ensuite séparer cette connexion.

Désactiver la carte IoT
La carte IoT n'a pas de coupe-circuit. Vous devez simplement séparer le câble USB du PC ou de l'alimentation électrique et la carte IoT se désactive. Le dernier programme sauvegardé démarre automatiquement à l'activation suivante. Le phénomène se répète lorsque l'on appuie sur la touche reset.

2 Désignée au cours de ce qui suit comme carte IoT.

3 La version 1.8.2 de l'Arduino IDE a été utilisée pour les projets. Si une nouvelle version devait être disponible avant la sortie du calendrier de l'Avent, vous pourrez l'utiliser. Si les modifications devaient être importantes, vous trouverez alors dans l'espace de téléchargement une indication à ce sujet.



Installation de l'Arduino IDE

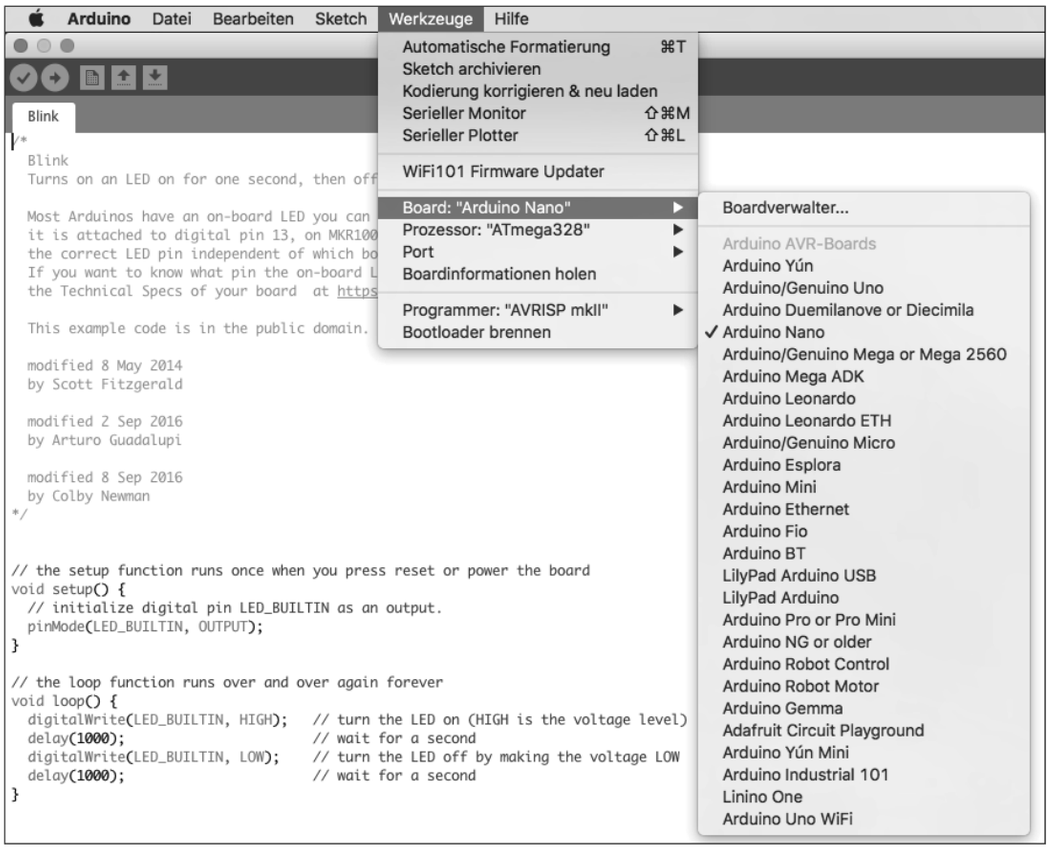
Téléchargez Windows Installer pour la version en cours Arduino IDE sur **www.arduino.cc/en/Main/Software** ou utilisez simplement le fichier `arduino-windows.exe` du téléchargement pour le calendrier de l'Avent. Sous Windows 10 vous pouvez télécharger et installer Arduino IDE également depuis le Windows Store.

Assurez-vous que tout soit coché dans la fenêtre de dialogue **Installation Options**. Selon la configuration Windows une confirmation du contrôle de compte utilisateur est nécessaire.

Mise à jour du firmware sur la carte IoT

Afin de pouvoir également comprendre tous les exemples avec votre carte IoT, mettez d'abord à jour le firmware de la carte. Pour cela, ouvrez d'abord l'Arduino IDE installé précédemment et choisissez dans le menu de l'Arduino IDE **Outils/Port**. Ici dans la plupart des cas seul un port en série unique s'affiche. Mettez la coche.

Choisissez ensuite dans l'option de menu **Outils/Carte** l'option **Arduino Nano**, si celle-ci n'a pas été détectée automatiquement. Vous devez sélectionner **ATmega328** comme processeur.



Choisir la carte adaptée dans Arduino IDE

Démarrage rapide du téléchargement du firmware sur la carte

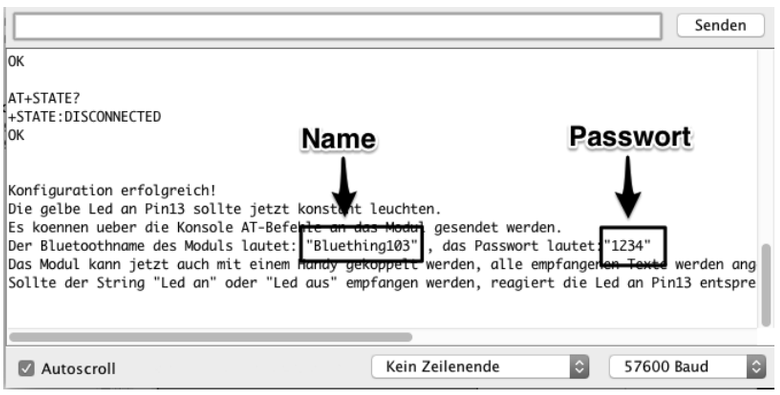
Après le démarrage de l'Arduino IDE et la sélection de la carte, ouvrez maintenant via l'option de menu **Fichier/Ouvrir...** le firmware **Firmware_V1.2b.ino**. Celui-ci se trouve dans l'archive de téléchargement dans le répertoire **Firmware_V1.2b**. Avant d'exécuter le firmware, assurez-vous que le jumper sur la carte IoT soit sur **AT**. Chargez le firmware sur la carte avec **Sketch/Uploader** Ou bien vous pouvez également utiliser l'icône prévue dans l'éditeur. Vous chargez le firmware4 sur la carte par la flèche à droite.

Après une courte période le message **Upload terminé** devrait apparaître dans la barre de statut de l'éditeur. La carte IoT est désormais mise à jour.

4 Le firmware est un programme Arduino normal. Ce type de programme Arduino est désigné par Sketch (ébauche).

Tester la carte IoT

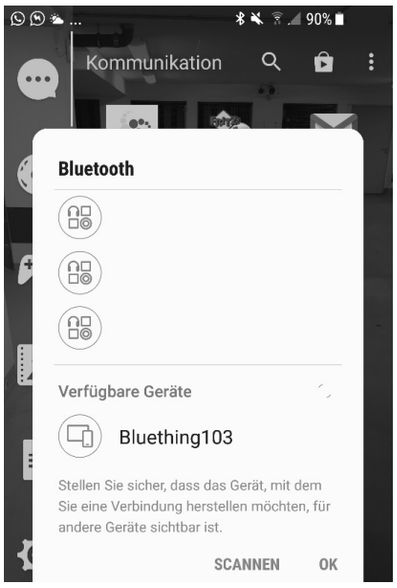
Vous pouvez tester la carte après exécution de la mise à jour du firmware. Ouvrez pour cela le moniteur sériel de l'Arduino IDE via **Outils/Moniteur sériel**. Paramétrez le transfert de données sur **57 600 Baud**. Dans la fenêtre de sortie du texte lisible doit maintenant apparaître . La sortie **configuration effectuée avec succès !** termine l'opération et est suivie d'informations pour le réseau sans fil.



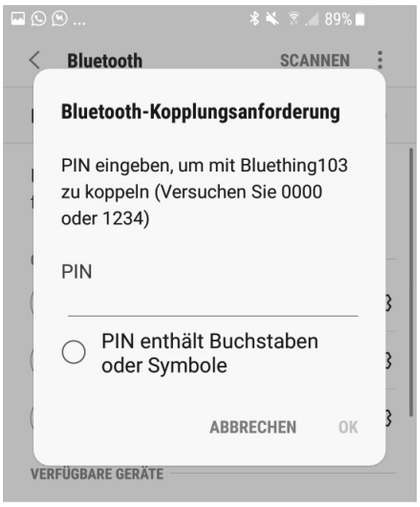
Le nom du réseau dans cet exemple est **Bluething103** et le mot de passe **1234**.

Maintenant vous pouvez tester avec votre Smartphone Android si vous voyez également le réseau sans fil. Pour cela activez le Bluetooth sur votre Smartphone et après un court moment le réseau sans fil doit apparaître. Sélectionnez maintenant le réseau sans fil (dans la capture d'écran suivante **Bluething103**) et confirmez la sélection par **OK**.

Le nom de la carte IoT sur votre système commence également par Bluething mais finit éventuellement par un autre numéro. Le numéro est généré individuellement avec l'adresse MAC de votre carte.



Le nom de la carte IoT sur votre système commence également par Bluething mais finit éventuellement par un autre numéro. Le numéro est généré individuellement avec l'adresse MAC de votre carte.



Vous devez saisir un mot de passe pour la connexion avec la carte IoT.

Après la saisie du bon mot de passe le réseau apparaît sous **APPAREILS COUPLÉS**. Maintenant votre Smartphone peut communiquer avec la carte IoT.

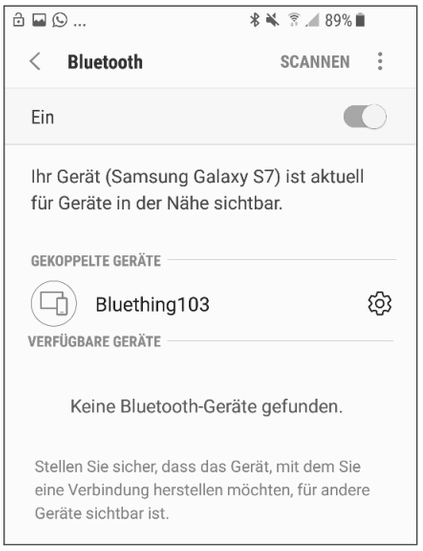
Faire clignoter la DEL On-Board

Il faut encore tester la programmation de la carte IoT. Comme exemple on utilise le Blink-Sketch intégré dans l'arduino IDE. Choisissez donc dans le menu **Fichier/Exemples/0.1 Basics/ Blink**.

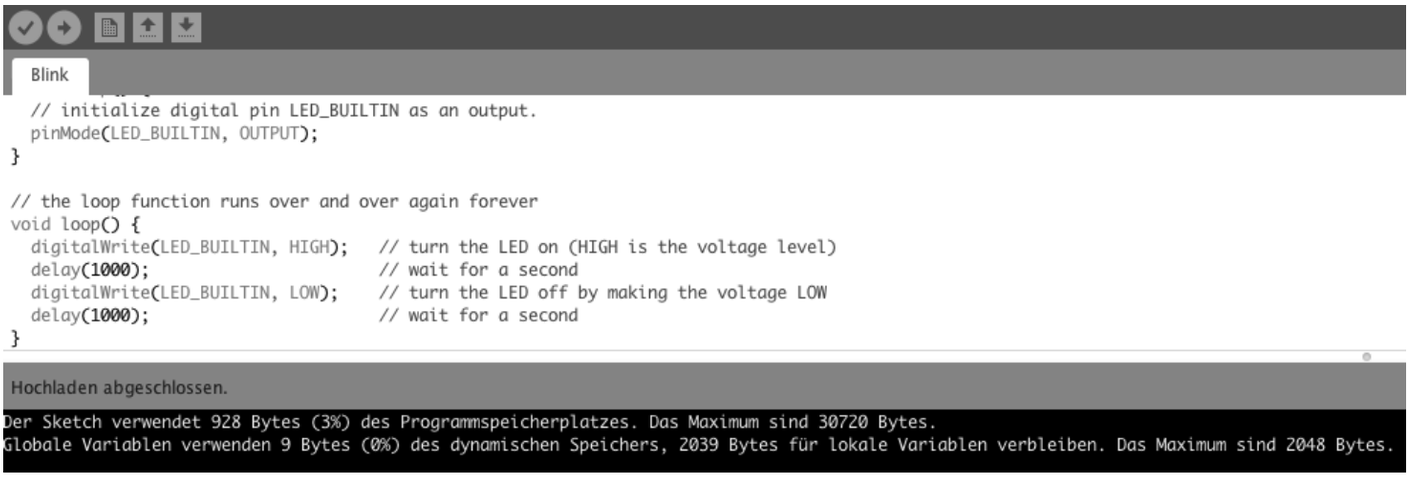
Cliquez maintenant en haut à gauche sur le symbole rond avec la flèche pour uploader le programme sur la carte IoT branchée, ce qu'on appelle aussi flash.

Vous compilez les données par la coche, vous chargez le programme sur la carte IoT avec la flèche vers la droite.

Une fois l'upload terminé, la DEL D2 (près du raccordement D13) clignote sur la carte IoT. Maintenant la carte est prête à fonctionner pour les jours à venir.



Dès qu'un appareil est couplé avec un Smartphone, une connexion est également possible sans renouveler le balayage.



En bas de la fenêtre du code source, on voit les sorties de l'Arduino IDE lors de la compilation et de l'upload.

Si la DEL ne clignote pas, regardez le message d'erreur dans l'Arduino IDE.

Ici la connexion à la carte IoT Board n'a pas marché. Dans ce cas le mauvais port a été sélectionné, on peut y remédier rapidement par l'option de menu **Outils/Port**.

Toujours utiliser la carte IoT en mode AT
Sur la carte IoT se trouve un jumper. Ce jumper doit être pour pour tous les projets dans ce calendrier sur **AT**.

2e jour

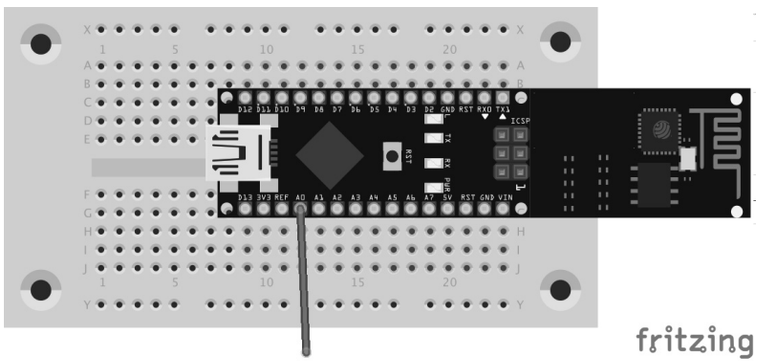
Aujourd'hui dans le calendrier de l'avent

- 1 Breadboard (SYB 46)
- 1 câble jumper

Mesurer des valeurs analogiques

Aujourd'hui vous programmez un Sketch dans l'Arduino IDE pour lire des valeurs d'une entrée analogique. Les valeurs sont représentées par sortie texte et graphiquement.

Composants: 1 Breadboard, 1 câble jumper (mâle - mâle)



Le câble jumper sert d'antenne. Essayer les mesures également sans ce câble pour voir si le câble a une influence sur votre environnement de mesure.

Le programme

Le programme de ce jour s'appelle `Jour02.ino` et se trouve dans le répertoire `Jour02` à l'intérieur de l'archive de téléchargement.

```
int analogValue = 0;
int analogPin = A0;

void setup() {
 pinMode(analogPin, INPUT);
 Serial.begin(9600);
}
```

```
void loop() {
  analogValue = analogRead(analogPin);
  Serial.println(analogValue);
  delay(1000);
}
```

Ainsi fonctionne le programme

```
int analogPin = A0;
```

Pour démarrer le numéro Pin utilisé dans la variable `analogPin` est défini. Tous les programmes dans l'Arduino IDE comportent au moins les deux fonctions `setup` et `loop` :

```
void setup() {
 pinMode(analogPin, INPUT);
 Serial.begin(9600);
}
```

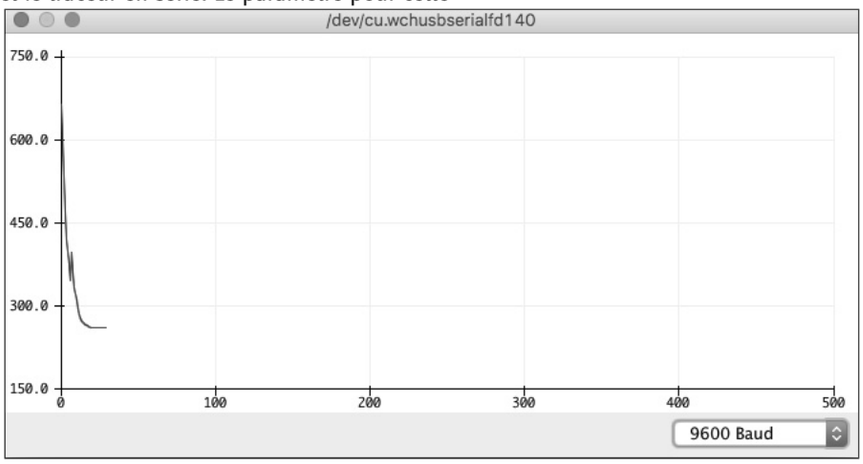
La fonction `setup` fonctionne une fois au démarrage et est utilisée la plupart du temps pour la configuration. Le pin analogique est défini comme entrée. La communication en série est lancée par `Serial.begin(9600)` pour indiquer les valeurs sur le moniteur et le traceur en série. Le paramètre pour cette fonction est la vitesse de transmission.

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

La fonction `loop` est répétée jusqu'à ce que l'on coupe l'alimentation électrique ou que l'on appuie sur le bouton reset.

```
void loop() {
  analogValue = analogRead(analogPin);
  Serial.println(analogValue);
  delay(1000);
}
```

La valeur de l'entrée analogique dans la variable `analogValue` est enregistrée par `analogRead` et sortie par `Serial.println`. Avec `delay(1000)` le programme attend 1 000 millisecondes.



Le débit de données doit être correctement paramétré dans le menu déroulant en bas dans le coin droit de la fenêtre, à savoir dans ce cas **9 600 Baud**.

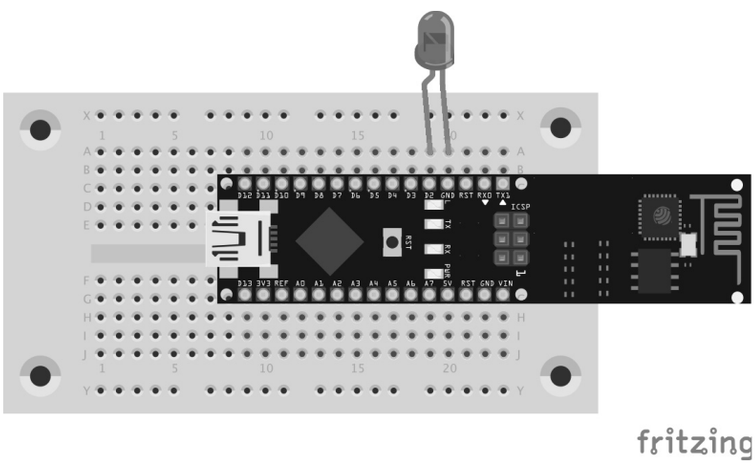
Ouvrez maintenant le moniteur en série via **Outils/Moniteur en série**. Vous voyez ainsi les valeurs mesurées. Vous pouvez également afficher les valeurs sous forme graphique. Pour cela fermez le moniteur en série ouvert précédemment et ouvrez le traceur en série via **Outils/Traceur en série**.

3. jour

3e jour

Aujourd'hui dans le calendrier de l'avent

- 1 DEL rouge avec résistance en série
- 1 fil de connexion



Vous n'avez pas besoin de résistance séparée, car il est déjà intégré dans la DEL.

Clignotant

Aujourd'hui vous allez faire clignoter une DEL à fréquence 2 Hz.

Composants: 1 Breadboard, 1 DEL rouge avec résistance en série

Le programme

Le programme de ce jour s'appelle `Jour03.ino` et se trouve dans le répertoire `Jour03`.

```
const int ledPin =  2;
int ledState = LOW;

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

void loop() {

    if (ledState == LOW) ledState = HIGH;
    else ledState = LOW;
    digitalWrite(ledPin, ledState);
    delay(500);
}
```

Ainsi fonctionne le programme

```
if (ledState == LOW) ledState = HIGH;
else ledState = LOW;
```

Dans la variable `ledState` on définit le clignotement ou non de la DEL. Pour commencer la variable a la valeur `LOW`. Cette valeur est changée toutes les 500 ms par `delay(500)`. De ce fait, la DEL clignote.

4e jour

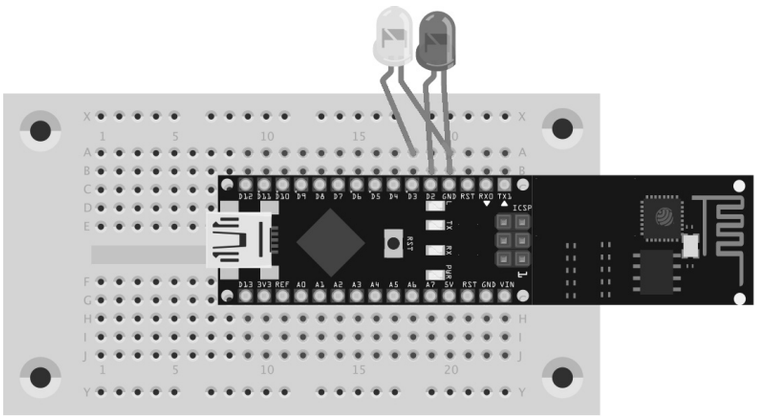
Aujourd'hui dans le calendrier de l'avent

· 1 DEL jaune avec résistance en série

Clignotant alternatif

Deux DELs clignotent en alternance.

Composants: 1 Breadboard, 1 DEL jaune avec résistance en série, 1 DEL rouge avec résistance en série



Les deux cathodes (jambe courte de la DEL) doivent être placées sur le GND. À gauche se trouve la DEL jaune, à droite la DEL rouge.

Le programme

Le programme de ce jour s'appelle `Jour04.ino` et se trouve dans le répertoire `Jour04`.

```
const int ledPin1 =  10;
const int ledPin2 =  12;

int ledState = LOW;
int pin = ledPin1;

void setup() {
  pinMode(ledPin1, OUTPUT);
     pinMode(ledPin2, OUTPUT);
  digitalWrite(ledPin1, LOW);
  digitalWrite(ledPin2, LOW);
}

void loop() {
    if (pin == ledPin1) {
      pin = ledPin2;
      digitalWrite(ledPin1, LOW);
    } else {
      pin = ledPin1;
      digitalWrite(ledPin2, LOW);
    }
    digitalWrite(pin, HIGH);
    delay(500);
}
```

Ainsi fonctionne le programme

```
    if (pin == ledPin1) {
     pin = ledPin2;
      digitalWrite(ledPin1, LOW);
    } else {
      pin = ledPin1;
      digitalWrite(ledPin2, LOW);
    }
```

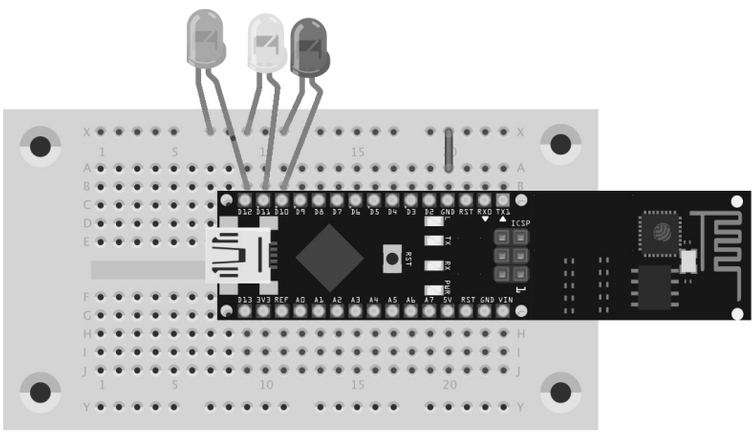
La DEL qui vient de s'allumer est enregistrée dans la variable temporaire `pin`. Initialement cette valeur est sur `ledPin1`. La variable est placée sur l'autre broche par une requête if et la DEL qui vient de s'éclairer est éteinte. La commutation a lieu toutes les 500 ms.

5. jour

5e jour

Aujourd'hui dans le calendrier de l'avent

- 1 DEL verte avec résistance en série



À partir de trois DELs, cela devient un peu plus étroit, par conséquent Le GND est spécialement placé sur la barre supérieure. DELs de gauche à droite : verte, jaune et rouge.

Feu

Le projet d'aujourd'hui est un feu créé avec l'environnement de développement graphique Snap!

Composants: 1 Breadboard, 1 DEL verte avec résistance en série, 1 DEL jaune avec résistance en série, 1 DEL rouge avec résistance en série, 1 fil de liaison

Installer Snap! et préparer la carte IoT

Snap! est un environnement de développement graphique qui existe également spécialement pour Arduino sous la forme de Snap4Arduino. Pour cela téléchargez la version du logiciel utilisée dans ce calendrier sous **http://www.buch.cd**5. Après l'installation changez la langue de la commande pour le français. Pour cela cliquez dans Snap4Arduino sur le symbole des paramétrages (engrenage) et sélectionnez dans le menu **Language** la langue **Français**.

Dans la catégorie **Arduino** se trouvent les éléments pour la commande de la carte IoT.

Pour pouvoir utiliser maintenant Snap4Arduino avec votre carte, vous devez installer un sketch spécial sur la carte, à savoir Firmata. Raccordez pour cela la carte IoT à votre PC et ouvrez l'Arduino IDE. Sélectionnez sous **Fichier/Exemples/Firmata/ Standard Firmata** et chargez le sketch en cliquant sur la flèche à droite de la carte. Maintenant vous pouvez programmer la carte avec Snap4Arduino.

Ne pas utiliser Arduino IDE et Snap! en même temps
L'Arduino IDE et Snap! ne peuvent pas être utilisés simultanément. Par conséquent fermez toujours l'un des programmes avant de travailler dans l'environnement souhaité.

Le programme créé avec Snap!

Basculer le programme dans Snap!

Un programme dans Snap! est composé d'éléments graphiques qui sont répartis en catégories. Le menu des catégories se trouve en haut à gauche.

Maintenant tirez hors de la **commande** l'élément avec la couleur verte au centre de la zone de travail. Pour déclencher et arrêter la DEL, utilisez l'élément **Régler la broche numérique**. Définissez les valeurs par l'élément commutable **vrai** des **opérateurs**. Pour que le programme ne se termine pas, vous avez besoin de l'élément à boucle de **contrôle**. Pour que les DELS ne basculent pas également immédiatement, établissez une pause d'une seconde par **pause** du **contrôle**.

Avant de pouvoir démarrer le programme, vous devez connecter la carte IoT avec Snap! Pour cela, cliquez sur l'élément **Connecter avec Arduino** dans la catégorie **Arduino**. En cliquant sur le symbole les raccordements disponibles sont affichés. Sélectionnez le premier raccordement. Maintenant cliquez sur la flèche verte (en haut à droite), et le feu commence déjà.

Ouvrir un projet externe dans Snap!
Le programme d'aujourd'hui se trouve dans le dossier **Jour05**. Dans la barre de menu, allez sur le premier symbole et choisissez **Importer...**. Naviguez dans le dossier **Jour05** et sélectionnez **jour05-snap.xml**. Le projet est ouvert maintenant et vous pouvez l'utiliser.

5 Ou bien vous pouvez également télécharger la dernière version sous **http://snap4arduino.org/**. La commande peut légèrement varier de la capture d'écran, les programmes devraient toutefois fonctionner.

6e jour

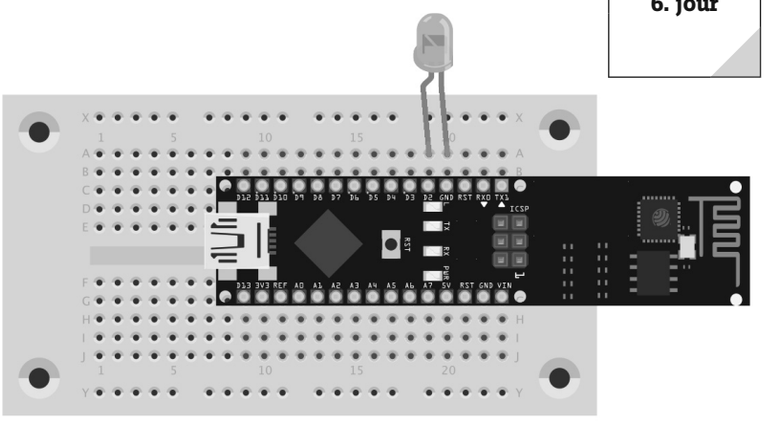
Aujourd'hui dans le calendrier de l'avent

- 1 DEL bleue avec résistance en série

Connexion avec la carte IoT

Aujourd'hui vous réalisez une connexion à la carte IoT avec votre Smartphone et vous enclenchez et arrêtez la DEL bleue par le Smartphone.

Composants: 1 Breadboard, 1 DEL bleue avec résistance en série



Le circuit ressemble au circuit du jour 3.

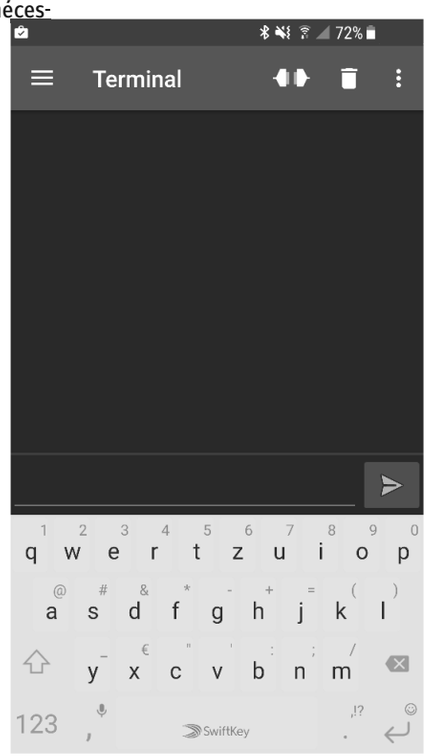
Installer l'appli pour le contrôle

Aujourd'hui vous n'avez pas encore besoin d'appli propre mais de contrôler la carte avec l'appli gratuite **Serial Bluetooth Terminal** de l'App-Store Google Play.

Le programme

Le programme d'aujourd'hui repose sur le firmware pour la carte IoT. Les parties de programmes nécessaires du firmware sont dans le fichier `Modèle.ino` dans le répertoire `Modèle`.

Connexion radio prête
Le firmware (et ainsi également le modèle) est programmé de telle sorte que la DEL orange sur la broche 13 s'allume en continu, dès que la connexion radio est prête. Par conséquent attendez jusqu'à ce que cette DEL s'allume avant de mettre en place une connexion avec la carte IoT.



Après le premier démarrage une fenêtre noire vide apparaît.

Copiez le fichier Modèle.ino dans un nouveau répertoire `Jour06` et renommez le fichier dans `Jour06.ino`. Ou bien vous pouvez utiliser directement le fichier fini `Jour06.ino` du répertoire `Jour06`. Maintenant ouvrez le fichier avec l'Arduino IDE. Le modèle contient déjà quelques fonctionnalités. La constante pour la DEL interne est déjà présente. Vous définissez la constante `LedPinBlue` pour la DEL supplémentaire directement sous la DEL interne :

```
#define LedPin 13
#define LedPinBlue 2
```

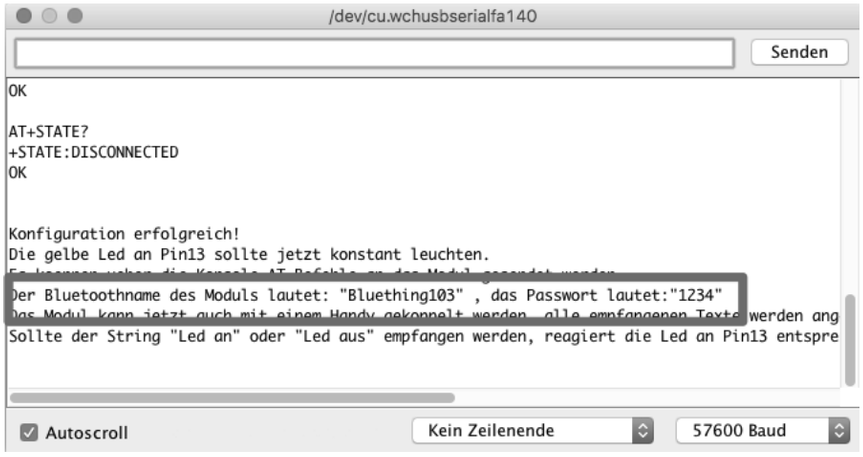
À la fin de la méthode `setup` la DEL est enclenchée :

```
void setup() {
  ...
  digitalWrite(LedPinBlue, HIGH);
}
```

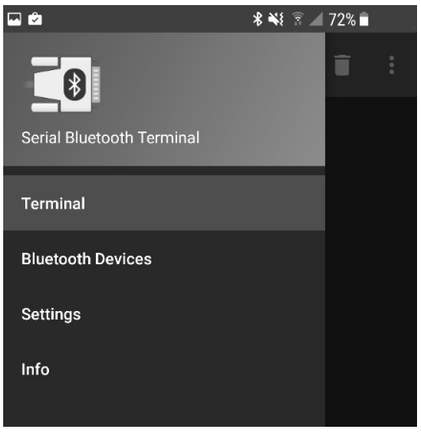
Pour l'enclenchement et l'arrêt de la DEL intégrée sur la broche 13 il y a déjà le code. Celui est maintenant étendu comme suit :

```
if (Text.startsWith(« Del allumée ») || Text.startsWith(« DEL allumée ») || Text.
startsWith(« DEL ALLUMÉE »)){
    digitalWrite(LedPin, HIGH);
    digitalWrite(LedPinBlue, HIGH);

  }
  if (Text.startsWith(« Del éteinte ») || Text.startsWith(« DEL éteinte ») || Text.
startsWith(« DEL ÉTEINTE »)){
    digitalWrite(LedPin, LOW);
    digitalWrite(LedPinBlue, LOW);
  }
```

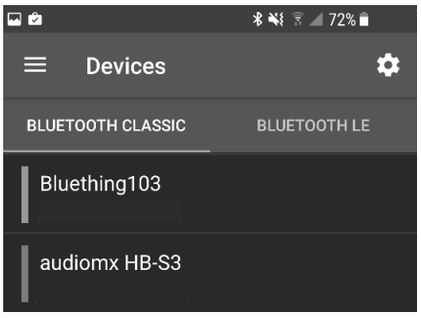
Si la carte reçoit `DEL allumée` ou `Del allumée` ou `DEL ALLUMÉE`, alors la DEL interne et la DEL bleurs sont allumées via `digitalWrite(LedPin, HIGH)` et `digitalWrite(LedPinBlue, HIGH)` .



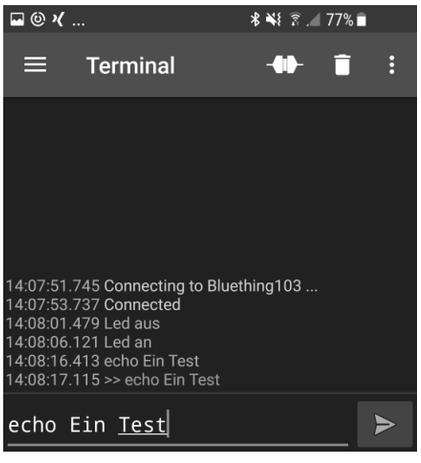
Le nom du réseau est **Bluething103** et le mot de passe **1234**.



Vous accédez aux paramètres de l'appli via le menu central.



Dans la liste des réseaux Bluetooth se trouve également le réseau de la carte IoT.



Vous pouvez voir via **echo** que l'appli peut également recevoir des informations de la carte IoT.

Si la carte IoT reçoit des données, la boucle suivante est exécutée :

```
while(HC05.available() > 0){
 …
 Text="";
}
```

Insérez maintenant avant la dernière ligne un appel de la fonction encore à programmer `hookRec(Text)` :

```
while(HC05.available() > 0){
 …
 hookRec(Text);
 Text="";
}
```

Dans cette fonction vous distribuez maintenant le texte reçu par l'interface sans fil :

```
void hookRec(String text) {
 if (text.startsWith("echo") || text.startsWith("Echo")) {
 HC05.print(">> " + text);
 }
}
```

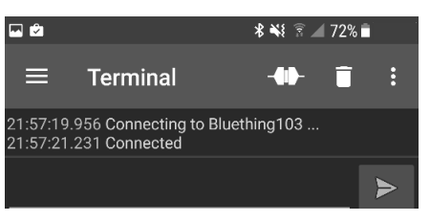
Maintenant vous pouvez faire tourner le programme sur la carte IoT. Activez le Bluetooth sur votre smartphone et sélectionnez le réseau Bluetooth nouvellement créé. Vous pouvez voir les noms correspondants dans le moniteur en série de l'Arduino IDE. Ouvrez la fenêtre correspondante via **Outils/Moniteur en série**. Lors de la connexion vous devez saisir un mot de passe, vous voyez également ce mot de passe dans le moniteur en série.

Maintenant lancez l'appli Bluetooth installée précédemment. Après le démarrage allez dans le menu de l'appli (trois traits superposés sur le côté gauche) et sélectionnez l'option de menu **Bluetooth Devices**.

Maintenant sélectionnez le réseau correspondant et passer dans le terminal (borne) en cliquant une fois sur l'icône du menu puis sélectionnez **Terminal**.

Dans le terminal cliquez sur le symbole de connexion (à gauche à côté de la corbeille). Après un laps de temps la saisie du mot de passe est demandée. Il s'agit de **1234**. La première fois l'établissement de la connexion ne fonctionne pas encore, cliquez donc de nouveau sur le symbole. Maintenant la connexion doit être établie. Dans la fenêtre du terminal le message **Connected** apparaît. Maintenant vous pouvez communiquer avec la carte IoT.

Double établissement de connexion nécessaire
Tant que vous n'avez pas encore saisi de mot de passe dans l'appli Bluetooth, vous devez établir la connexion deux fois. Saisissez le mot de passe la première fois et la deuxième fois la connexion sera établie.



L'appli terminal est maintenant prête pour la communication.

Si vous entrez maintenant **Del éteinte** et appuyez sur le symbole de la flèche, les deux DELs sont éteintes. En entrant **Led allumée** vous rallumez les DELs. Chaque entrée qui commence par **echo** est envoyée sur la carte et celle-ci renvoie l'entrée commençant par deux flèches. Le texte est sorti par le terminal.

7e jour

Aujourd'hui dans le calendrier de l'avent

· 1 potentiomètre, 15 kΩ

Chenillard contrôlable

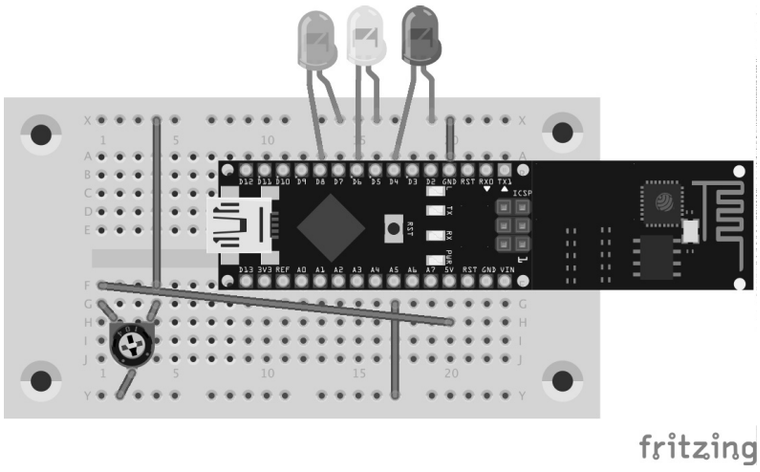
Le projet d'aujourd'hui est un chenillard dont la vitesse peut être contrôlée par un potentiomètre.

Composants: 1 Breadboard, 1 DEL verte avec résistance en série, 1 DEL jaune avec résistance en série, 1 DEL rouge avec résistance en série, 1 potentiomètre 15 kΩ, 4 fils de liaison (longueurs différentes)

Le programme

Le programme de ce jour s'appelle `Jour07.ino` et se trouve dans le répertoire `Jour07`.

```
int analogPin = A5;
int analogValue;

int led1 = 8;
int led2 = 6;
int led3 = 4;

void setup() {
 pinMode(analogPin, INPUT);

 pinMode(led1, OUTPUT);
 pinMode(led2, OUTPUT);
 pinMode(led3, OUTPUT);
 digitalWrite(led1, LOW);
 digitalWrite(led2, LOW);
 digitalWrite(led3, LOW);
 Serial.begin(9600);
}

void loop() {
  analogValue = analogRead(analogPin);
  Serial.println(analogValue);

  delay(analogValue);
  digitalWrite(led3, LOW);
  digitalWrite(led1, HIGH);

  delay(analogValue);
  digitalWrite(led1, LOW);
  digitalWrite(led2, HIGH);

  delay(analogValue);
  digitalWrite(led2, LOW);
  digitalWrite(led3, HIGH);
}
```



Le potentiomètre prend beaucoup d'espace occupé. C'est pourquoi il faut pousser la carte IoT vers le haut contrairement au jour précédent. DELs de gauche à droite : verte, jaune et rouge.

Ainsi fonctionne le programme

```
  analogValue = analogRead(analogPin);
  delay(analogValue);
  digitalWrite(led3, LOW);
  digitalWrite(led1, HIGH);
```

On peut lire la valeur configurée du potentiomètre dans la variable temporaire `analogPin`. `delay` fait attendre et ensuite les DELs sont allumées.

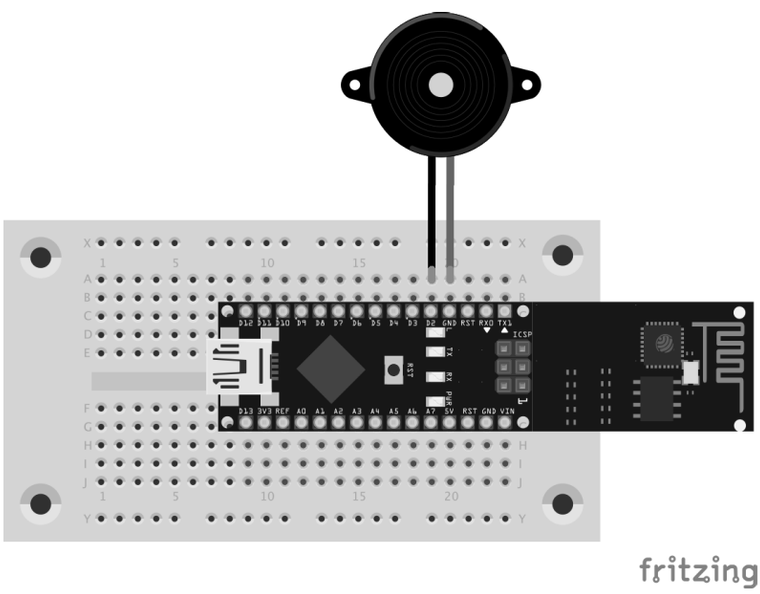
8. jour

8e jour

Aujourd'hui dans le calendrier de l'avent
- 1 Piezo

Émettre du son via une appli
Émettez du son sur un Piezo par une appli.

Composants: 1 Breadboard, 1 Piezo



Les deux fils du Piezo sont connectés avec D2 et GND. Le reste est résolu par le logiciel.

Environnement de développement pour les applis
Le contrôle de la carte IoT est réalisé dans ce calendrier de l'Avent par l'appli déjà utilisée **Serial Bluetooth Terminal** et avec les applis Smartphone développées pour le système d'exploitation Android6. Vous pouvez télécharger chaque fichier de projet fini sous **http://www.buch.cd**. MIT App Inventor 2 **(http://appinventor.mit.edu)** est utilisé comme environnement de développement.

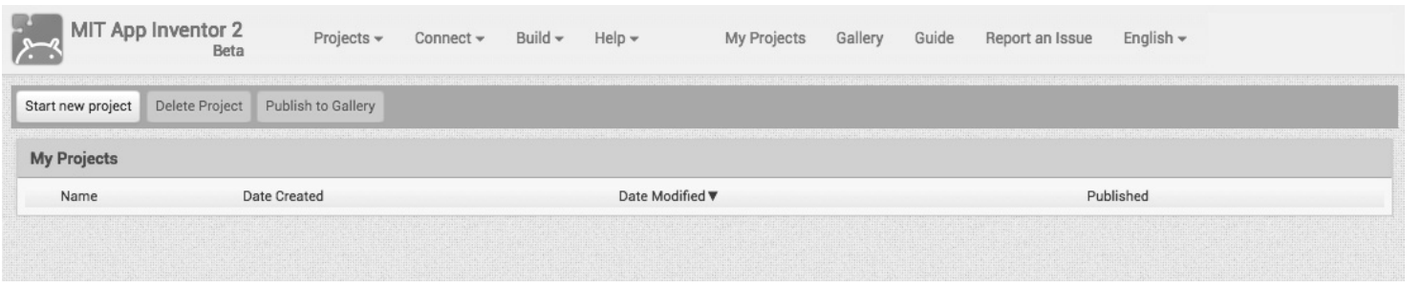
L'environnement de développement fonctionne dans le navigateur. C'est pourquoi vous avez besoin d'un accès internet pendant le temps de développement.

En cliquant sur le bouton **Create apps!** vous démarrez l'environnement de développement. Il vous faut un compte gratuit sur Google pour l'utiliser.

Créer un compte sur Google
Si vous n'avez pas encore de compte sur Google, créez un pour le développement avec **MIT App Inventor 2 (AI2)** un compte via **https://accounts.google.com**.

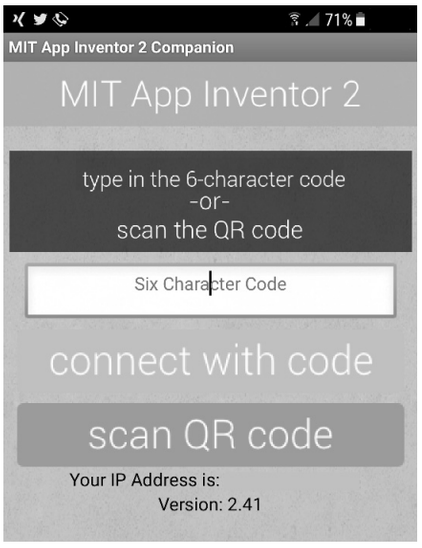
Après une inscription réussie, vous êtes dans le tableau de bord. Toutes vos applis développées jusqu'à présent sont indiquées ici. Comme vous n'avez probablement pas encore développé d'appli avec AI2, le tableau de bord est vide.

6 Le chipset intégré sur la carte IoT n'est supporté que par les Smartphones Android.



La commande n'est donc pas encore disponible en français.

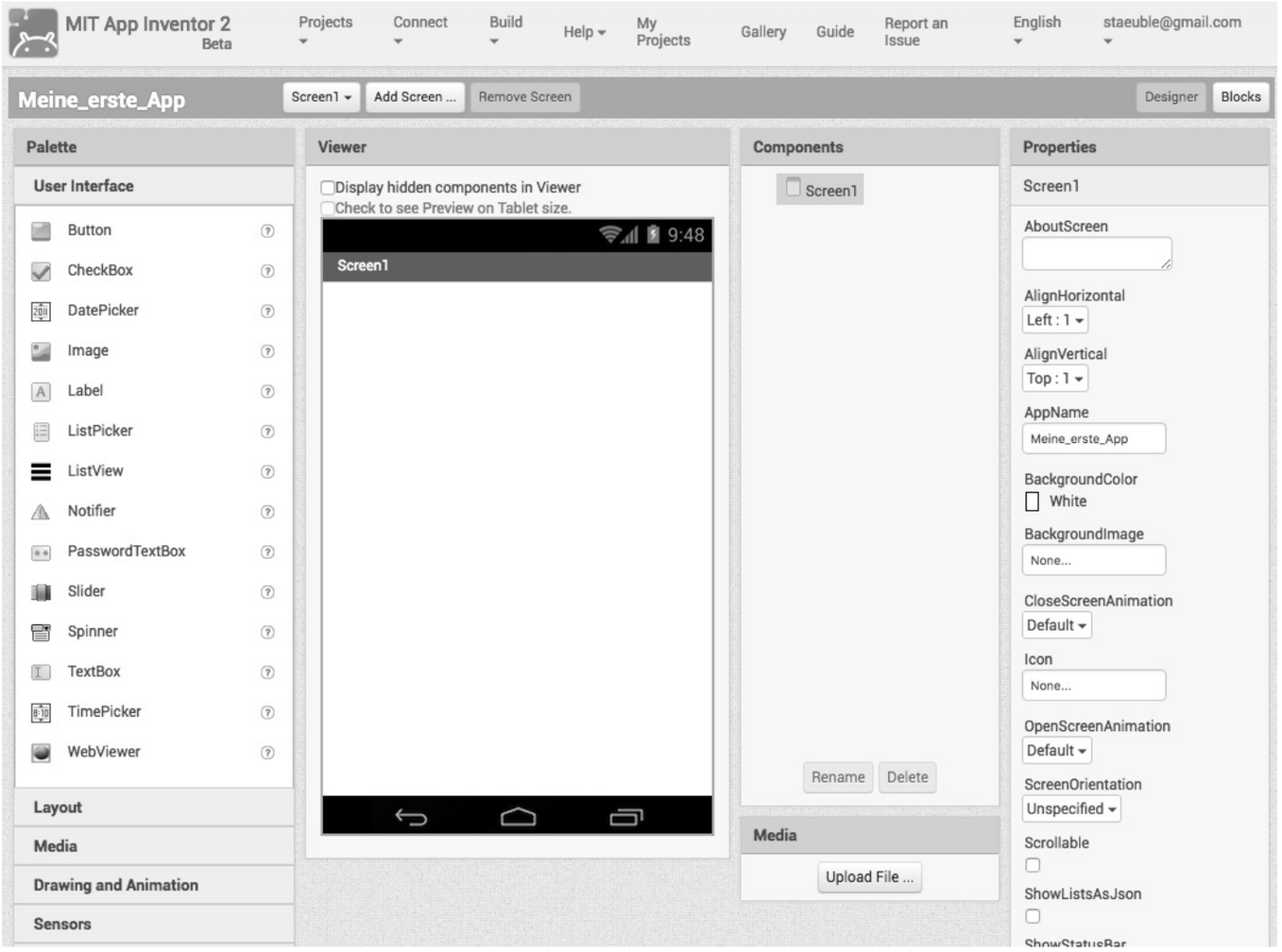
Pour pouvoir tester rapidement l'appli que vous avez développée également pendant le développement, vous devez installer l'appli gratuite **MIT AI2 Companion** du Google Play Store sur votre Smartphone. Après le démarrage vous devez saisir le code de votre appli, pour cela vous devez d'abord créer une appli.



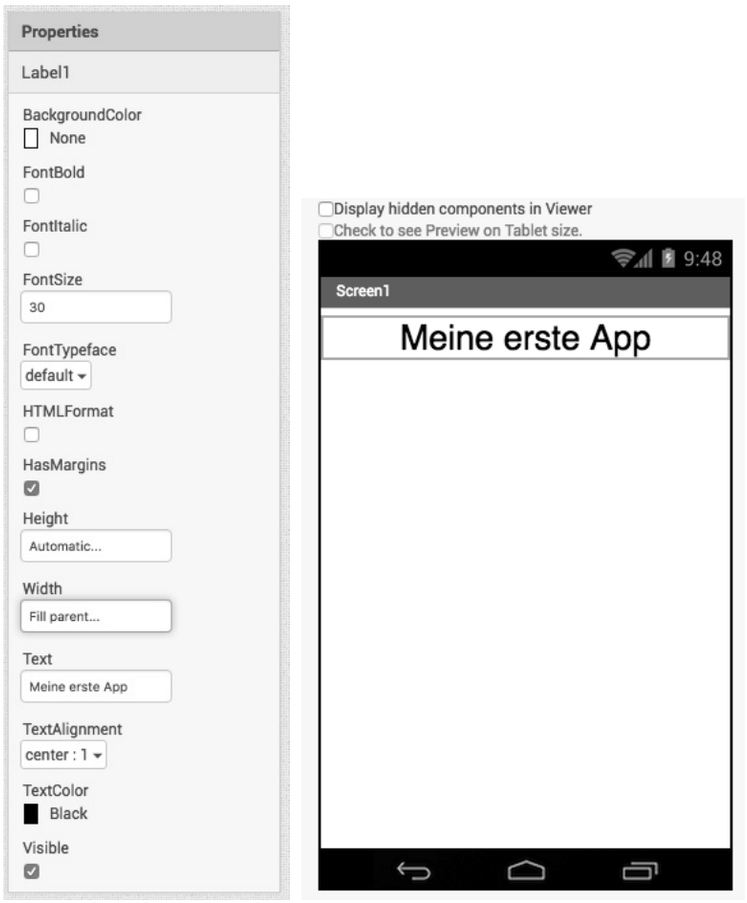
Pendant le développement ne lancez pas directement l'appli mais passez par le biais de l'appli MIT Inventor 2 Companion. Vous vous épargnez ainsi une installation manuelle de l'appli sur l'appareil.

La première appli avec AI2

Cliquez maintenant sur la commande de démarrage **Start new project** et saisissez comme nom **Ma_première_appli**, veillez à ce que le nom ne comprenne aucun espace vide. Maintenant ouvrez l'environnement de développement dans votre navigateur.



Après le démarrage, l'interface de l'appli est encore vide.



Procédez aux modifications d'un composant dans la fenêtre **Properties**. L'affichage dans Viewer est modifié immédiatement.

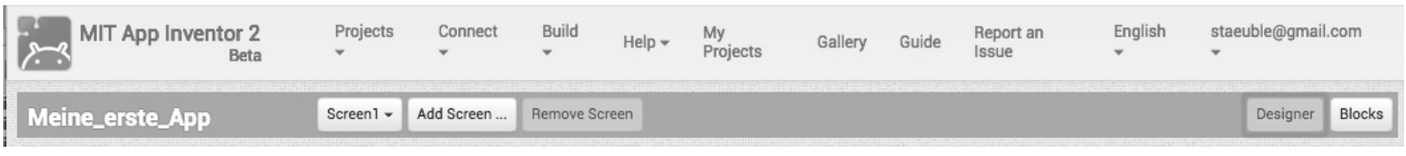
La fenêtre est partagée en plusieurs zones. À gauche vous voyez dans la zone **Palette** les éléments graphiques disponibles pour votre appli. À côté vous voyez dans **Viewer** l'interface de l'appli. Dans **Components** vous visualisez les composants utilisés dans votre appli et dans **Properties** les propriétés des composants en cours de sélection.

Maintenant faites glisser un marquage sous la forme d'une étiquette dans la fenêtre. Ajustez le marquage en plaçant **Width** sur **Fill parent** et **TextAlignment** sur **center:l** Maintenant le marquage est centré horizontalement. Ensuite modifiez la taille de police dans **FontSize 30** et placez le texte (champ : **Text)** sur **Ma première appli**.

Maintenant, l'application doit être étendue à une interaction. Pour ce faire, ajoutez un bouton dans la fenêtre **Components** et modifier via **Properties**la légende dans **Touch me**. Notamment ajoutez une étiquette avec l'inscription **Status: not touched**.

Interface de l'appli avec les trois composants

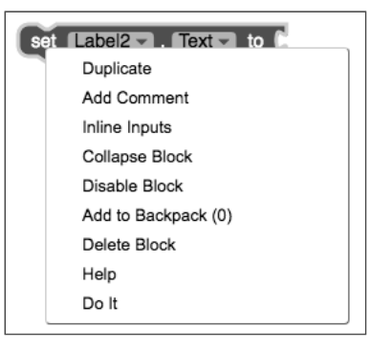
AI2 a deux affichages pour une appli : l'affichage pour la conception de l'interface, que l'on appelle designer (onglet **Designer)**, et l'affichage pour la programmation dans la langue du bloc (onglet **Blocks**). On voit ces deux onglets dans la zone supérieure de l'environnement de développement.



Vous accédez aux deux affichages de l'environnement de développement par les onglets **Designer** et **Blocks**.

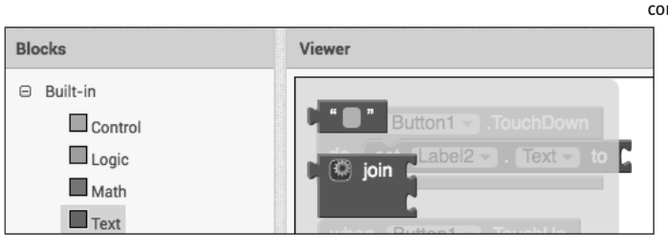
Après avoir cliqué sur **Blocks** l'affichage de la langue de programmation des blocs s'ouvre. Sélectionnez maintenant à l'intérieur de la zone **Blocks** les composants **Button1**, et vous voyez déjà les événements disponibles auxquels vous pouvez répondre dans l'appli.

Sélectionnez maintenant les deux blocs **when Button1.TouchDown** et **when Button1.TouchUp**. Dans les deux cas le contenu de l'étiquette **Label2** doit être ajusté. Pour cela cliquez sur **Label2** et sélectionnez l'élément **set Label2.Text to**. Soit vous ajoutez l'élément deux fois ou bien une fois et vous copiez l'élément par le menu contextuel (clic souris droit) des composants.

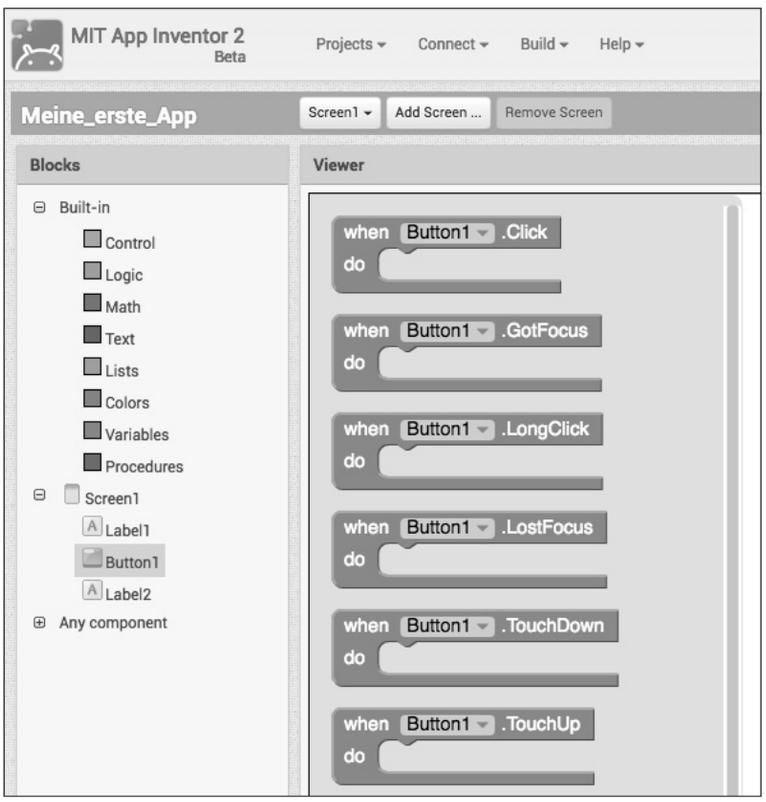


Les composants ont un menu contextuel Copiez les composants sélectionnés via **Duplicate**

Insérez les deux composants maintenant dans les deux composants à l'intérieur des deux éléments ajoutés précédemment (**when Button1.TouchDown** et **when Button1.TouchUp**). Il manque maintenu le contenu texte. Pour cela sélectionnez l'élément vide (élément supérieur) de la catégorie **Text**.



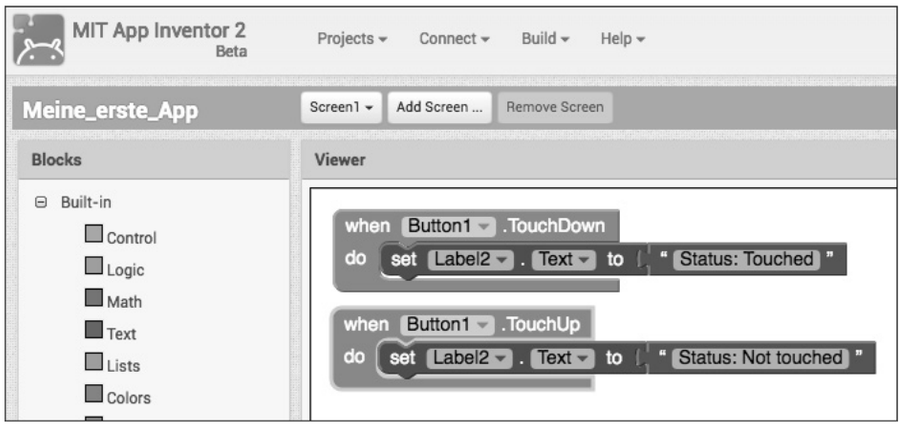
L'élément entre guillemets est utilisé pour modifier l'inscription.



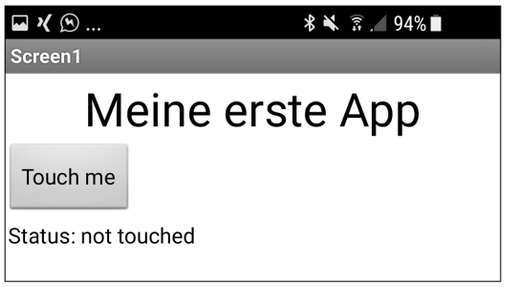
Pour programmer la logique, vous pouvez aller à la sélection par catégories (**Built-in**) ou par composants (**Screen1**).

Dans le cas de **when Button1.TouchDown** le texte est placé sur **Status: Touched**. Pour **when Button1.TouchUp** le texte est placé sur **Status: Not touched**.

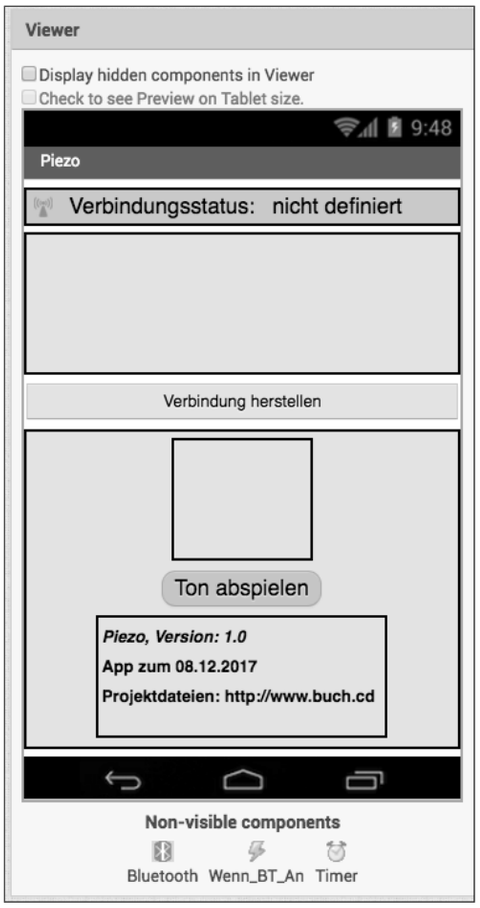
Maintenant vous pouvez tester l'appli sur votre Smartphone. Sélectionnez pour cela dans le menu **Connect/AI Companion**. Maintenant un code textuel et un code QR apparaissent. Saisissez maintenant soit le code dans **MIT AI2 Companion** ou scannez le code QR via **scan QR code**. Vous démarrez l'appli avec **connect with Code**. L'appli s'ouvre maintenant sur votre Smartphone. Lorsque vous touchez maintenant le bouton, le texte se modifie.



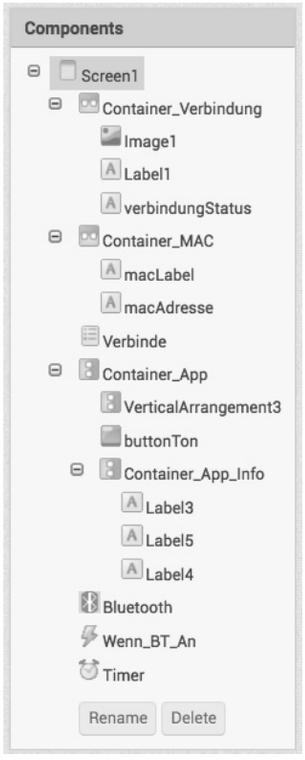
Le texte de l'étiquette est ajusté en touchant le bouton.



Développer l'appli avec AI2 est aussi simple que ça.



Dans la zone supérieure de l'appli est indiqué si la connexion Bluetooth est établie ou non.



Tous les composants contenus dans l'appli dans un aperçu hierarchisé. Les composants importants ont été dotés d'un nom correspondant.

Contrôler le piezo avec l'appli

Pour contrôler la carte IoT par une appli, vous avez besoin d'un sketch sur la carte, qui réagit aux ordres de l'appli, et d'une appli. Aujourd'hui le piezo joint doit être contrôlé par une appli. Le sketch nécessaire est `Jour08.ino` dans le répertoire `Jour08`. Chargez ce sketch sur la carte IoT.

Le sketch est basé sur les données déjà utilisées `Modèle.ino`. Le principe de tous les sketchs dans ce calendrier de l'Avent répondant à une appli est le suivant : un texte est envoyé par l'interface radio, celui-ci est analysé dans le sketch, et ensuite une action est exécutée. Cela fonctionne exactement ainsi à l'inverse : l'appli reçoit un texte de la carte IoT par l'interface radio, analyse le texte et ensuite une action correspondante est exécutée. L'ordre pour jouer un son est `Son` :

```
if (Text.startsWith(“Son”) || Text.startsWith(“SON”) ||   Text.
startsWith(“Son”)){
        playMelody();
   }
```

L'appli correspondante se trouve dans le fichier `Piezo.aia`. Importez le fichier dans AI2.

Importer les projets dans AI2
Toutes les applis préfabriquées sont disponibles dans l'archive téléchargement comme fichiers aia. Pour ouvrir un fichier dans AI2, vous devez importer le fichier projet par l'option de menu **Projects / Import project (.aia) from my computer ...**.

Fonctionnalités de l'appli

Les applis suivantes qui ont été créées avec AI2 sont établies avec un modèle, par conséquent cette appli sera décrite dans le détail et il y sera fait référence dans la suite du déroulement. L'appli est composée en tout de quatre zones :

- Zone de connexion
- Zone fonctionnelle
- Zone info de l'appli
- Zone interne non visible

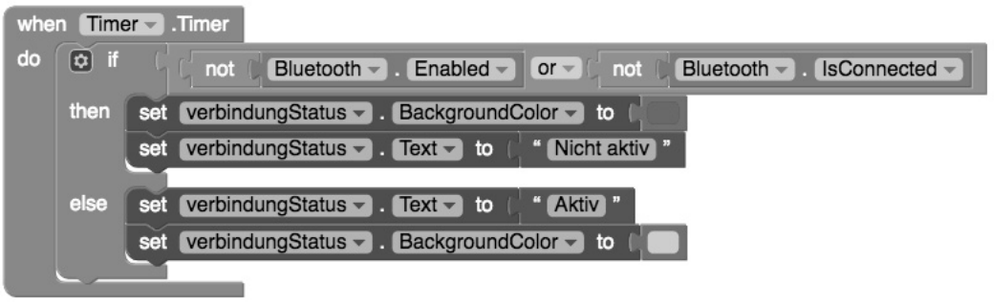
On peut voir les zones individuelles de l'appli dans la figure de gauche. **Container_Connexion** et **Container_MAC** font partie de la zone de connexion. L'application réelle qui varie également d'un projet à l'autre, est située dans **Container_App**. Ce container comprend la zone fonctionnelle **(buttonTon)** et la zone info de l'appli **(Container_App_Info)**.

Qu'est-ce qu'un container ?
Pour le placement des éléments vous disposez dans la catégorie **Layout** de plusieurs gestionnaires de disposition (p. ex. **HorizontalArrangement)** ; ceux-ci vous aident pour le placement horizontal et vertical. Les éléments sont placés à l'intérieur d'un gestionnaire de disposition de ce type. Comme ces gestionnaires de disposition contiennent souvent plusieurs éléments, ils sont désignés comme container (Layoutcontainer).

Modifier le nom des composants
Lorsque vous ajoutez des composants à une interface dans AI2, un nom est automatiquement généré à ces composants. Contrairement aux autres propriétés d'un composant, vous ne pouvez pas modifier le nom par la fenêtre **Properties** mais dans la zone **Components**. Ici vous sélectionnez les composants correspondants et cliquez sur le bouton **Rename**.

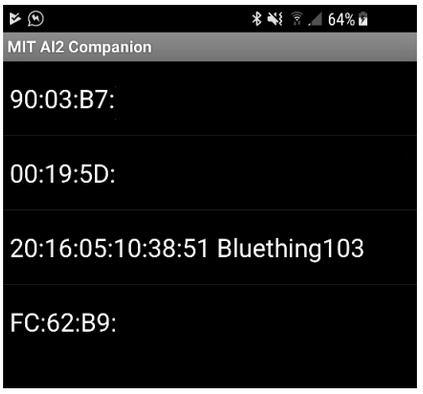
L'interface comprend en tout trois composants non visibles. Pour utiliser le Bluetooth vous avez besoin des composants **BluetoothClient** de la catégorie **Connectivity**. Le composant est uniquement placé dans la zone de l'interface et ne doit pas être configuré. Pour sélectionner le dispositif Bluetooth correspondant il faut le composant **ActivityStarter** (image en haut**Si_BT_Sur)** de **Connectivity** La valeur `android.bluetooth.adapter.action.REQUEST_ENABLE` est attribuée à l'ActivityStarter dans la fenêtre **Properties** . Pour le test permanent de la connexion radio, l'élément **Clock** (dans l'image en haut **Minuterie)** des **Capteurs** est utilisé. Le **TimerInterval** est fixé à **200** dans la fenêtre **Properties**.

La programmation se fait dans l'onglet **Blocs**. Le statut de connexion est fixé par le **Timer**. Vous pouvez contrôler si l'appli est connectée à un dispositif Bluetooth par la propriété **isConnected**.



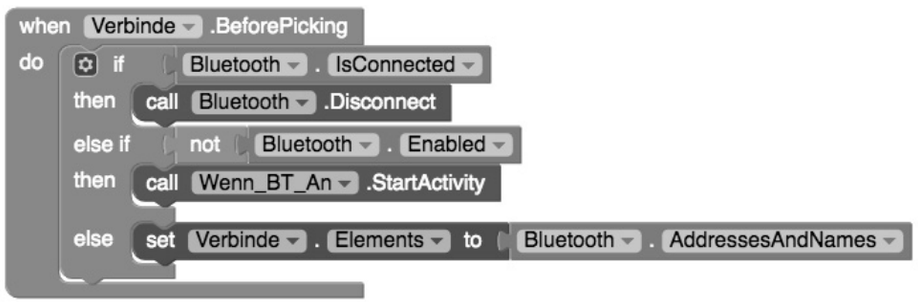
La connexion sans fil est contrôlée régulièrement.

En cliquant sur le bouton **Établir la connexion** la liste des dispositifs Bluetooth s'affiche. Si la liste est vide, vous devez retourner dans l'appli, alors l'autorisation d'accéder à l'interface Bluetooth est automatiquement demandée. Saisissez **1234** comme mot de passe. En cliquant de nouveau sur le bouton la liste devrait désormais être représentée.



Le nom et l'adresse MAC des périphériques disponibles sont affichés.

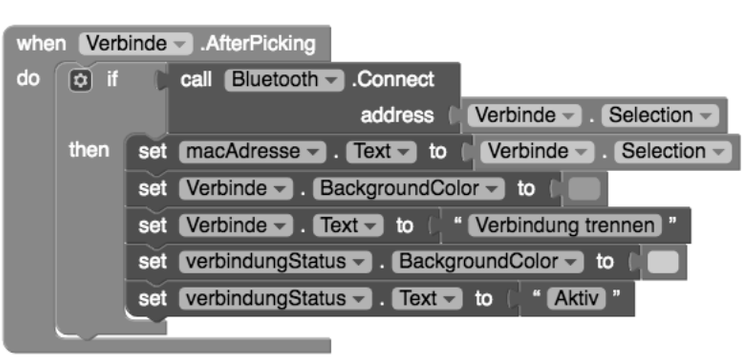
Ici, une entrée avec le nom **Bluething** devrait apparaître. Après avoir cliqué sur l'entrée vous devez encore saisir le mot de passe **1234** , et l'appli est connectée à la carte. L'affichage de cette liste est réalisé par le bloc **when Verbinde.BeforePicking**. Comme vous le voyez sur le bloc suivant, le bouton a deux fonctions : se connecter et se déconnecter.



La connexion Bluetooth peut être initiée entre la carte et l'appli par ce bouton.

La connexion elle-même est créée dans le bloc **when connect. AfterPicking**

Les trois blocs ci-dessus sont présents dans toutes les applications du calendrier de l'Avent. La véritable fonctionnalité de l'appli d'aujourd'hui est réalisée dans le bloc **when buttonSon.Click** Dans cette fonction seule la valeur **Son** est transmise, fermée par un `\n`. L'envoi d'un texte par l'interface sans fil est réalisé par le bloc **call Bluetooth.SendText**. Comme paramètres, vous passez le texte. N'oubliez pas de définir un critère de fin. Dans cet exemple le critère de fin est `\n`. Le sketch répond à ce caractère.



La connexion avec le périphérique sélectionné est créée.

Copier des blocs dans d'autres projets
Dans la zone **Blocs** vous voyez un sac à dos, appelé Backpack. Par ce moyen vous pouvez copier des blocs depuis l'éditeur de blocs d'un projet dans un autre projet. Tirez simplement les blocs correspondants dans le Backpack et vous pouvez réutiliser de suite les blocs dans un autre projet. Pour vider à nouveau le Backpack, faites un clic souris droit dans la zone de travail et choisissez **Empty Backpack**.

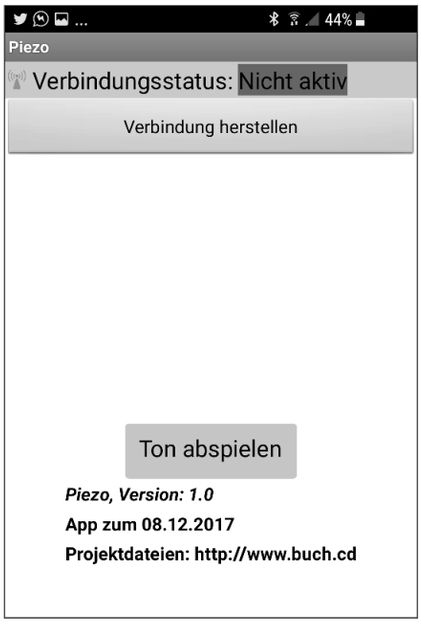
Les chaînes peuvent être liées entre elles par le composant **join**.

Tester l'appli

Lancez l'appli maintenant avec **Connect/AI Companion** et appelez l'appli **MIT AI2 Companion** sur votre Smartphone android. Scannez le code QR ou saisissez le code et lancez l'appli par **connect with code**. L'appli démarre maintenant.

Touchez maintenant **Établir la connexion**. Maintenant vous pouvez vous connecter avec la carte IoT.

Lorsque que vous touchez maintenant **Jouer un son**, une mélodie est jouée sur la carte IoT par le piezo.



Au début la connexion n'est pas encore activée.

L'appli est maintenant connectée avec la carte IoT.

9e jour

Aujourd'hui dans le calendrier de l'avent

· 1 DEL RVB avec résistance en série

DELs RVB

Une DEL normale allume toujours une seule couleur. Les DELs RVB utilisées dans le calendrier de l'Avent peuvent allumer au choix plusieurs couleurs. Ici dans le principe trois DELs de différentes couleurs sont intégrées dans un boîtier transparent. Chacune de ces trois DELs a une propre anode par laquelle elle est connectée à une broche GPIO. La cathode qui est connectée au fil de terre n'intervient qu'une seule fois. C'est pourquoi une DEL RVB a quatre fils de raccordement.

Broches de raccordement d'une DEL RVB

Les fils de raccordement des DELs RVB ont des longueurs différentes pour être reconnus clairement. Contrairement à la DEL normale, la cathode a ici le fil le plus long.

Les DELS RVB fonctionnent comme trois DELs individuelles et nécessitent par conséquent également trois résistances en série de 220 ohms. Celles-ci sont déjà incluses dans les DELs RVB de ce calendrier de l'Avent.

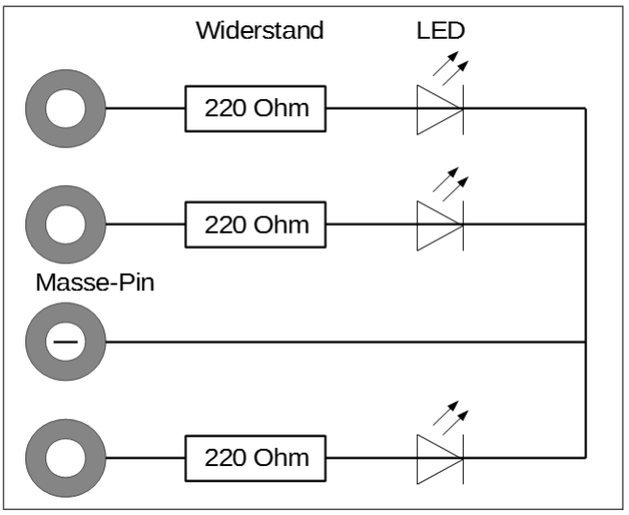
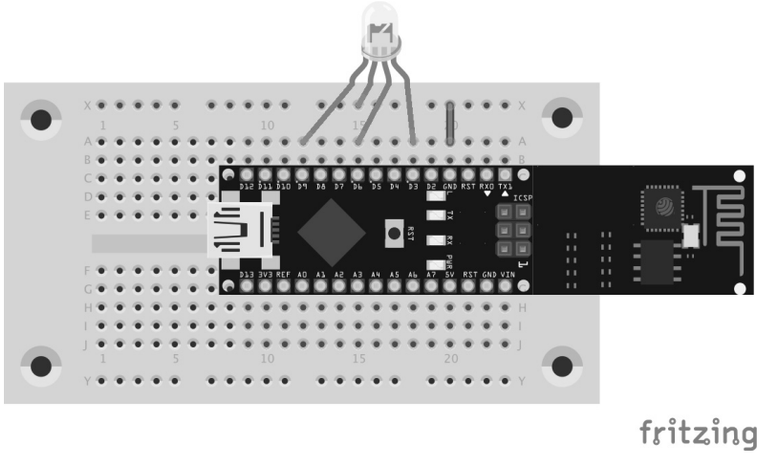


Schéma électrique d'une DEL RVB avec 3 résistances en série

Changer la couleur d'une DEL RVB par l'appli

Aujourd'hui vous programmez une appli pour changer la couleur d'une DEL RVB.

Composants: 1 Breadboard, 1 DEL RVB avec résistance en série, 1 fil de liaison



Les résistances en série sont déjà intégrées à la DEL RVB. La deuxième jambe (petite jambe) est la cathode et doit être branchée à la terre.

Le Sketch pour la carte IoT

Le programme de ce jour s'appelle `Jour09.ino` et se trouve dans le répertoire `Jour09`. Le Sketch répond à quatre boutons :

```
if (Text.startsWith(« Rouge ») || Text.startsWith(« ROUGE ») || Text.startsWith(« rouge” »){
   rouge();
 } else if (Text.startsWith(« Bleu ») || Text.startsWith(« BLEU ») || Text.startsWith(« bleu »)){
   bleu();
 } else if (Text.startsWith(« Vert ») || Text.startsWith(« VERT ») || Text.startsWith(« vert »)){
   vert();
 } else if (Text.startsWith(« Arrêt ») || Text.startsWith(« ARRÊT ») || Text.startsWith(« arrêt »)){
   arrêt();
 }
```

Les DELs sont allumées par quatre boutons.

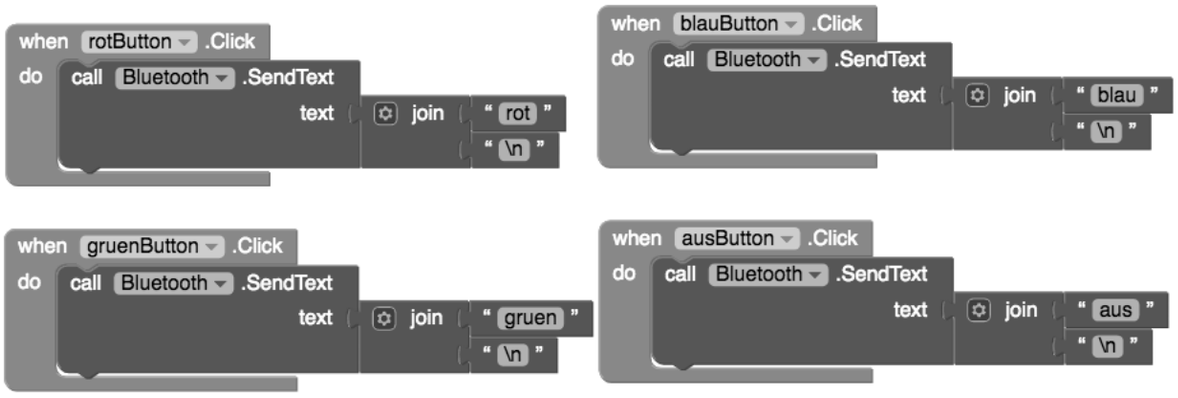
Chaque couleur a sa propre fonction dans le Sketch. Ensuite la fonction `rouge()` est introduite :

```
void rouge() {
  Serial.println(« Rouge »);
  analogWrite(redPin, HIGH);
  analogWrite(greenPin, LOW);
  analogWrite(bluePin, LOW);
}
```

L'appli

L'appli correspondante se trouve dans le fichier `RVB.aia`. Importez le fichier dans AI2.

Un texte correspondant est envoyé par l'interface Bluetooth indépendamment du bouton appuyé : Pour rouge `rouge\n` est envoyé, pour bleu `bleu\n` pour vert `vert\n` et pour l'arrêt `arrêt\n` .



Les blocs dans AI2

10e jour

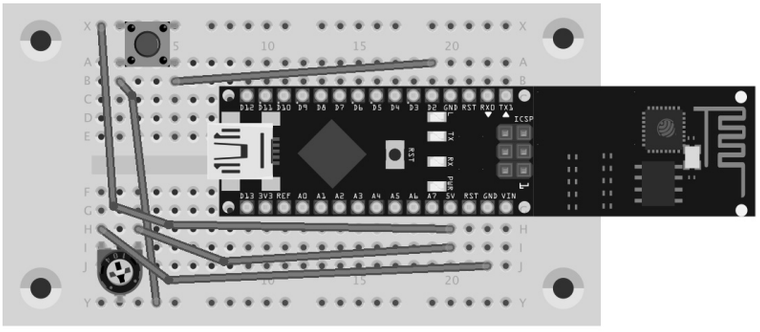
Aujourd'hui dans le calendrier de l'avent

· 1 poussoir

Afficher la pression de la touche

Aujourd'hui votre carte IoT répond à une frappe mécanique et transmet un message à l'interface radio.

Composants: 1 Breadboard, 1 poussoir, 1 potentiomètre, 5 fils de liaison (différentes longueurs)



La carte IoT est très longue, le poussoir doit donc être inséré en travers du Breadboard et les fils de liaison doivent être partiellement placés sous le câble USB branché.

Les broches numériques peuvent non seulement sortir des données, par exemple par des DELs, mais également être utilisées pour la saisie de données. Pour l'entrée nous utilisons dans le projet d'aujourd'hui un poussoir qui est directement enfiché sur la platine enfichable. Le poussoir comporte quatre broches de connexion, dans lesquelles deux opposées (longue distance) sont reliées entre elles. Tant que le poussoir est enfoncé tous les raccordements sont connectés entre eux. Contrairement à un interrupteur, un poussoir ne s'enclenche pas. La connexion est immédiatement arrêtée en relâchant le poussoir.

Si un signal +5 V se trouve sur une entrée numérique, celui-ci est analysé comme **vrai**.

Dans le cas d'un poussoir ouvert l'entrée n'aurait eu aucun état clairement défini. Lorsqu'un programme demande cette broche, cela peut entraîner des résultats aléatoires. Pour empêcher cela, on ferme une résistance comparativement très haute contre la terre. Cette résistance appelée déroulante (pull down) tire le statut de la broche d'entrée pour le poussoir ouvert de nouveau en bas sur 0 V. Comme la résistance est très élevée, il n'y a aucun risque de court-circuit tant que le poussoir est enfoncé. Lorsque le poussoir est enfoncé +5 V et le fil de terre sont directement connectés par cette résistance.

Le Sketch

Le programme de ce jour s'appelle `Jour10.ino` et se trouve dans le répertoire `Jour10`. L'appel de **digitalRead** analyse si le poussoir a été enfoncé :

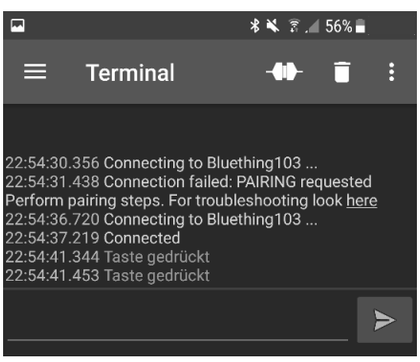
```
void loop() {
 int reading = digitalRead(buttonPin);

 if (reading != lastButtonState) {
    lastDebounceTime = millis();
  }

  if ((millis() - lastDebounceTime) > debounceDelay) {
    if (reading != buttonState) {
      buttonState = reading;

      if (buttonState == HIGH) {
        ledState = !ledState;
      }
    }
 }

 digitalWrite(LedPin, ledState);
```

Si le poussoir a été enfoncé, une notification est envoyée par l'interface radio :

```
if (reading != lastButtonState) {
 HC05.print(« Poussoir enfoncé\n »);
}
```

Afficher la réponse de la carte IoT

Pour l'affichage de la pression de la touche sur le Smartphone il faut utiliser l'appli **Serial Bluetooth Terminal** qui a déjà été utilisée. Après la connexion avec la carte IoT, pour une pression de la touche le message **Touche enfoncée** est affiché.



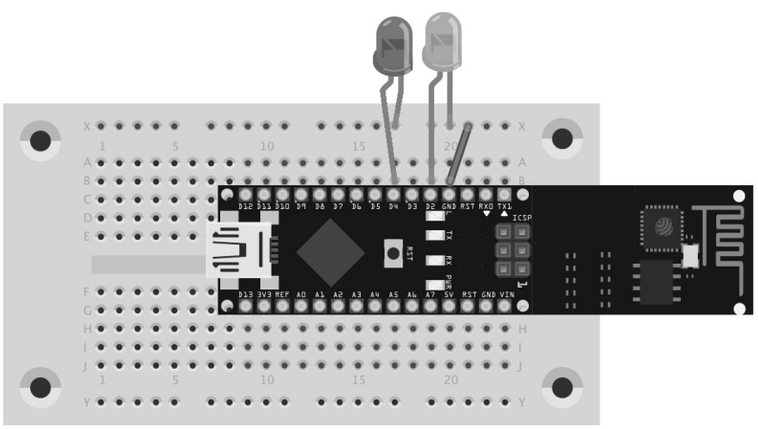
Les message reçus sont écrits en vert dans la fenêtre du terminal.

11. jour

11e jour

Aujourd'hui dans le calendrier de l'avent

- 1 fil de connexion



Le circuit est très compact par les résistances intégrées. DELs de gauche à droite : rouge et vert

DEL Écho par appli

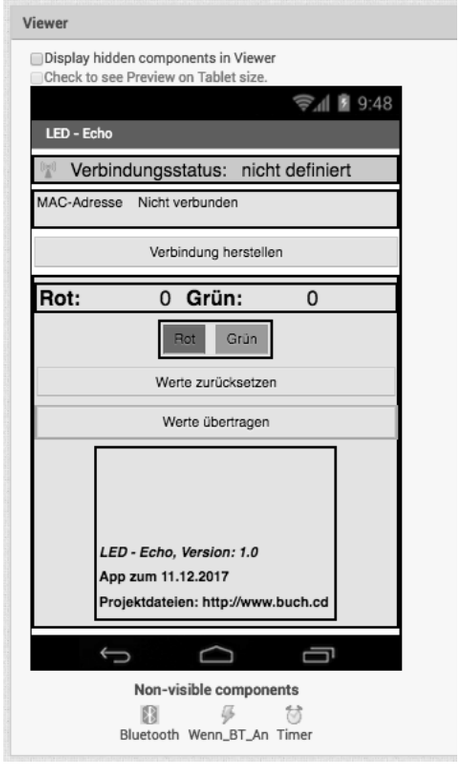
Le projet d'aujourd'hui est une DEL Écho Vous réglez une séquence par deux boutons dans l'appli et les DELs clignotent dans cette séquence sur le Breadboard.

Composants: 1 Breadboard, 1 DEL rouge avec résistance en série, 1 DEL verte avec résistance en série, 1 fil de liaison

Le Sketch

Le programme de ce jour est `Jour11.ino` et se trouve dans le répertoire `Jour11`. La séquence d'écho est transmise par l'appli. Le texte vient sous la forme suivante : RNNVNN. R est pour rouge et V pour vert, p. ex. R5V3 – ici la DEL rouge clignoterait cinq fois et ensuite la verte trois fois :

```
if (Text.indexOf(« R ») != -1 && Text.indexOf(« V »)
!= -1) {
 fixecouleur(texte);
} else if (Text.startsWith(« Arrêt ») || Text.
startsWith(« ARRÊT ») || Text.startsWith(« arrêt »)){
 arrêt();
}
```



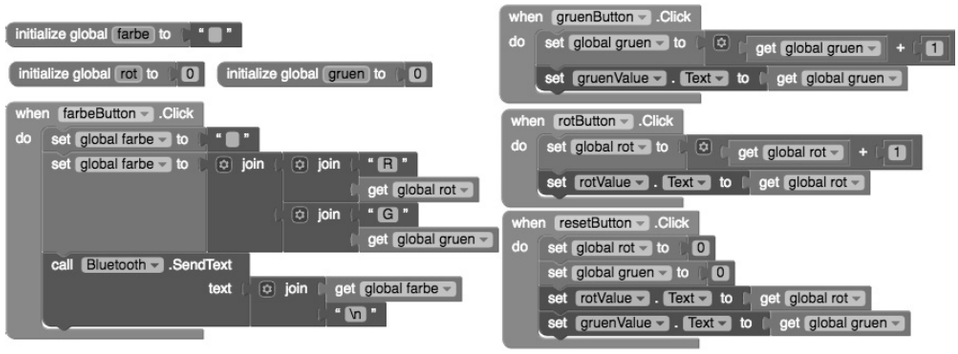
L'appli utilise déjà l'appli connue du jour 8.

L'appli

L'appli dispose de quatre boutons à côté du déjà connu **Établir la connexion** :

Rouge, **Vert**, **Transférer des valeurs** et **Restaurer les valeurs**.

Au contact de **Rouge** ou **Vert**, l'étiquette respective est incrémentée de un. Avec **Restaurer les valeurs** les valeurs sont remises à 0. Avec **Transférer les valeurs** les valeurs sont transmises à la carte IoT. La valeur à transmettre est mise en mémoire tampon dans une variable globale. Le contenu de la variable est transmis en Bluetooth en touchant **Transférer des valeurs**.



Pour chaque bouton il y a une requête when.Click.

12e jour

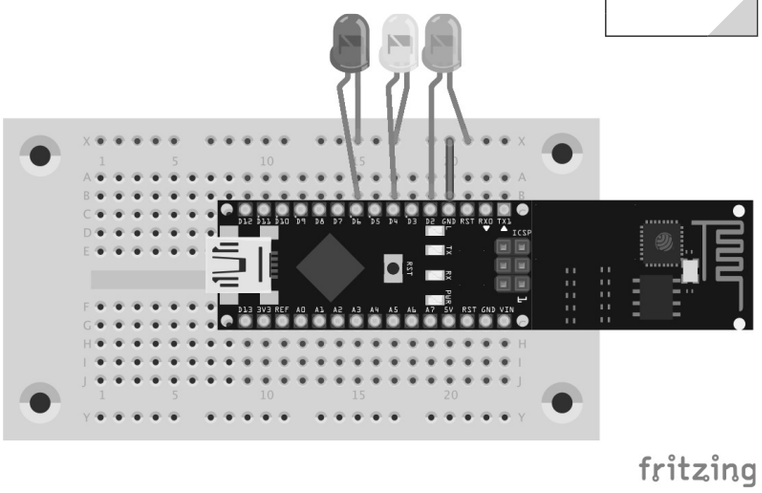
Aujourd'hui dans le calendrier de l'avent

- 1 DEL orange avec résistance en série

Régler la vitesse du chenillard par l'appli

Dans le projet d'aujourd'hui, vous contrôlez la vitesse d'un chenillard en Bluetooth.

Composants: 1 Breadboard, 1 DEL rouge avec résistance en série, 1 DEL orange avec résistance en série, 1 DEL verte avec résistance en série, 1 fil de liaison



Chenillard avec trois DELs : rouge, orange et vert.

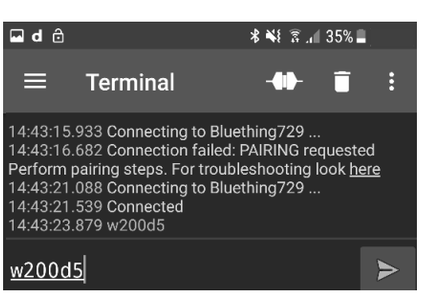
Le Sketch

Le programme de ce jour est `Jour12.ino` et se trouve dans le répertoire `Jour12`. Il faut deux paramètres pour contrôler le chenillard : le temps d'attente, pendant qu'une autre DEL est allumée, et le nombre d'itérations. Ces valeurs sont envoyées par l'appli en une chaîne sous la forme wTTTdNN. Par exemple w500d4 signifie que chaque DEL 500 ms s'allume et qu'il y a une attente de 500 ms avant l'activation d'une autre DEL. Le chenillard fonctionnerait en tout quatre fois :

```
if (Text.indexOf(“w”) != -1 && Text.indexOf(“d”) != -1) {
 String temp = Text.substring(Text.indexOf(“w”)+1,Text.indexOf(“d”));
 temps d’attente int =  temp.toInt();
 temp = Text.substring(Text.indexOf(“d”)+1);
 itérations int =  temp.toInt();
 chenillard(temps d’attente, itérations);
}
```

L'appli

L'appli **Serial Bluetooth Terminal** est utilisée pour la transmission des données. Après l'établissement de la connexion vous pouvez envoyer directement l'ordre à la carte IoT dans un format défini.



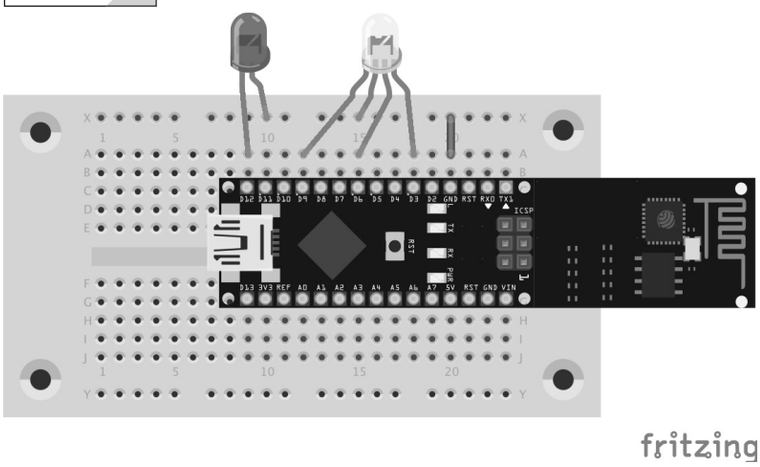
Dans l'émission on voit que deux tentatives de connexion sont nécessaires pour une connexion réussie avec la carte IoT. Si la connexion ne réussit pas, vérifiez par l'option de menu **Bluetooth Devices**,si vous avez sélectionnez le bon périphérique.

13. jour

13. jour

Aujourd'hui dans le calendrier de l'avent

- 1 DEL rose avec résistance en série



La DEL rose est utilisée pour signaler la connexion établie.

Adapter RVB par le slider dans l'appli

Une DEL RVB peut afficher non seulement l'une des trois couleurs primaires, mais aussi les transitions. Vous pouvez paramétrer précisément les couleurs de la DEL RVB au moyen d'une appli.

Composants: 1 Breadboard, 1 DEL RVB avec résistance en série, 1 DEL rose avec résistance en série, 1 fil de liaison

Le Sketch

Le programme de ce jour est `Jour13.ino` et se trouve dans le répertoire `Jour13`. Le contrôle de la DEL RVB est réalisé par une chaîne sous la forme RNNNVNNNBNNN. La couleur est fixée par la méthode `fixercouleur` seulement lorsque une chaîne est reçue par la carte IoT sous cette forme. Pour couper la DEL la chaîne `Arrêt` doit être envoyée :

```
while(HC05.available() > 0){
 Caractère = HC05.read();
 Text.concat(caractère);
 if (caractère == ‚\n‘) {
  if (Text.indexOf(“R”) != -1 && Text.indexOf(“G”) && Text.indexOf(“B”) ) {
   fixecouleur(texte);
  } else if (Text.startsWith(« Arrêt ») || Text.startsWith(« ARRÊT ») || Text.
startsWith(« arrêt »)){
    arrêt();
  }
  Text=””;
}
```

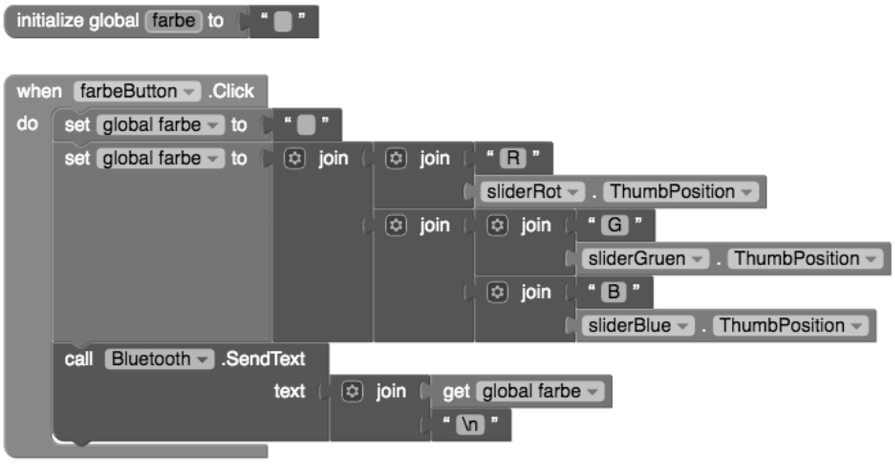
Dans la fonction `fixecouleur` le texte transmis avec la méthode `substring` est divisé en éléments et les valeurs sont ensuite écrites via `analogWrite` :

```
void fixecouleur(String text) {
  if (text.indexOf(“R”) != -1 && text.indexOf(“G”) && text.indexOf(“B”) ) {
    String temp = text.substring(text.indexOf(“R”)+1,text.indexOf(“G”));
    if (temp.indexOf(“.”) != -1) {
      temp = temp.substring(0, temp.indexOf(“.”));
    }
    int rouge =  temp.toInt();

    temp = text.substring(text.indexOf(“G”)+1,text.indexOf(“B”));
    if (temp.indexOf(“.”) != -1) {
      temp = temp.substring(0, temp.indexOf(“.”));
    }
    int vert =  temp.toInt();

    temp = text.substring(text.indexOf(“B”)+1);
    if (temp.indexOf(“.”) != -1) {
      temp = temp.substring(0, temp.indexOf(“.”));
    }
    int bleu =  temp.toInt();
    analogWrite(redPin,rouge);
    analogWrite(greenPin,vert);
    analogWrite(bluePin, bleu);
  }
}
```

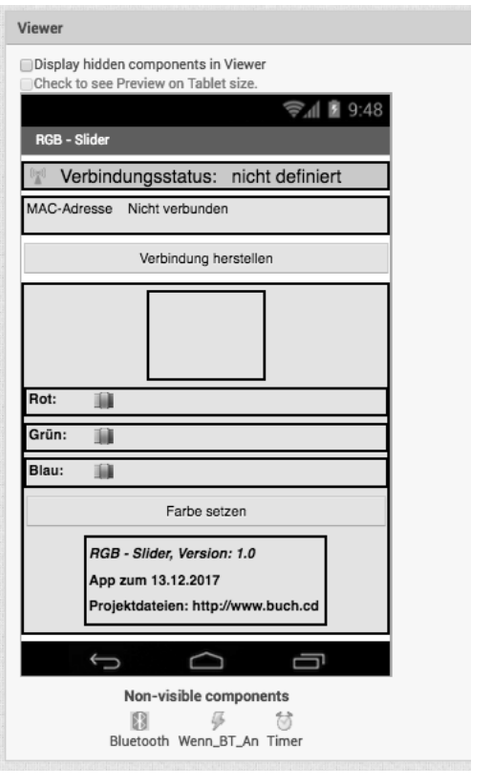
L'appli

AI2 est l'environnement de développement de l'appli d'aujourd'hui. Importer à cet effet le fichier `RVB_Slider.aia`. La valeur de chaque couleur est définie par trois sliders. En actionnant **Fixer la couleur** la valeur des trois sliders est envoyée à la carte IoT.

En touchant **fixer la couleur** le texte à transmettre est composé dans la variable `couleur`. Le texte est envoyé par l'interface sans fil via `Bluetooth.SendText`.



La composition de la chaîne est réalisée via **join** que l'on peut également mettre en paquets.



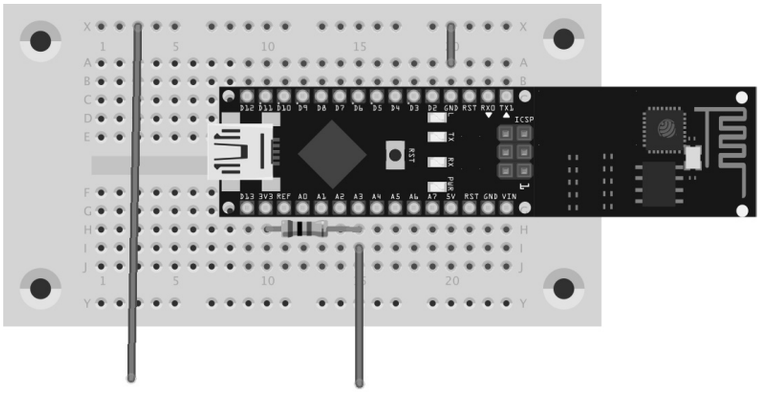
Chaque slider est en **HorizontalArrangement** avec une étiquette. Pour que les sliders soient l'un en dessous de l'autre la largeur des trois étiquettes a été fixée à 15 %.

14. jour

14e jour

Aujourd'hui dans le calendrier de l'avent

- 1 pâte à modeler rouge
- 1 potentiomètre, 20 MΩ



L'extrémité des deux fils de liaison va dans un morceau de pâte à modeler. Le mieux est de former pour chacun d'eux une petite boule ronde. Le contact de pâte à modeler sur A3 est utilisé comme entrée.

Contact de pâte à modeler

Aujourd'hui vous utilisez la pâte à modeler fournie comme entrée et vous modifiez en touchant la pâte à modeler la couleur de fond d'une surface dans l'appli.

Composants: 1 Breadboard, 2 morceaux de pâte à modeler, 1 résistance 20 MΩ, 3 fils de liaison (différentes longueurs)

C'est ainsi que fonctionnent les contacts de capteur

La broche activée comme entrée est connectée avec +3,3 V par une résistance à forte valeur ohmique (20 MΩ) de sorte à ce qu'il y ait un faible signal toutefois clairement défini comme élevé (high). Une personne qui ne plane pas librement dans l'air est toujours à la terre et fournit un niveau bas par la peau conductrice en électricité. Si cette personne touche maintenant un contact de capteur, le faible High signal est superposé par le niveau Low (bas) nettement plus puissant de la pointe du doigt et tire la broche sur le niveau Low.

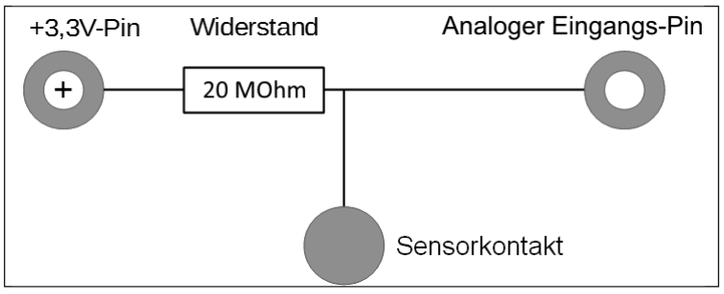


Schéma électrique pour les contacts de capteurs sur la carte IoT

La hauteur réelle de la résistance entre la main et la terre dépend de beaucoup d'éléments, entre autres chaussures et sol. Un pied nu dans de l'herbe humide est la meilleure connexion à la terre, toutefois cela fonctionne plutôt bien également sur du carrelage. Les parquets isolent plus, les revêtements de sol en plastique sont souvent chargés positivement. Pour que le circuit fonctionne toujours, un contact de masse est prévu en plus à chaque circuit comme pour les touches de détecteur sur les ascenseurs et les portes . Si la personne touche en même temps ceux-ci et le capteur, la connexion à la terre est établie dans tous les cas.

La pâte conduit le courant presque aussi bien que la peau humaine. Elle se modèle facilement dans des formes quelconques et une pâte de contact a un meilleur contact qu'un simple fil. La surface avec laquelle la main a touché le contact est nettement plus grande. Ainsi il est difficile d'avoir un « mauvais contact ». Coupez environ 10 cm de fil de connexion, retirez aux deux extrémités environ 1 cm d'isolant et enficher une extrémité dans un morceau de pâte. Enfichez l'autre extrémité comme sur l'illustration dans le Breadboard.

La carte IoT dispose d'entrées analogiques qui s'adaptent bien aux contacts de capteurs. Les entrées analogiques donnent des valeurs entre 0 (niveau bas) et 1 023 (haut niveau). Les valeurs entre 100 et 200 sont de bonnes valeurs limites pour différencier le contact de capteur touché et celui non touché.

Le Sketch

Le programme de ce jour est `Jour14.ino` et se trouve dans le répertoire `Jour14`. La véritable fonctionnalité dans ce sketch peut être programmée en peu de lignes. La valeur est lue avec `analogRead` et si celle-ci est inférieure à la valeur seuil, alors un texte est envoyé par l'interface avec `HC05.print`.

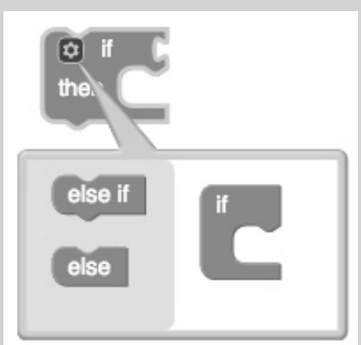
```
void loop() {
 sensorValue = analogRead(sensorPin);

 if (sensorValue < 256) {
    HC05.print("LOW\n");
    delay(1000);
  }

}
```

L'appli

L'appli d'aujourd'hui `Pâte_Interrupteur.aia` contient une étiquette dont la couleur de fond au début est orange.

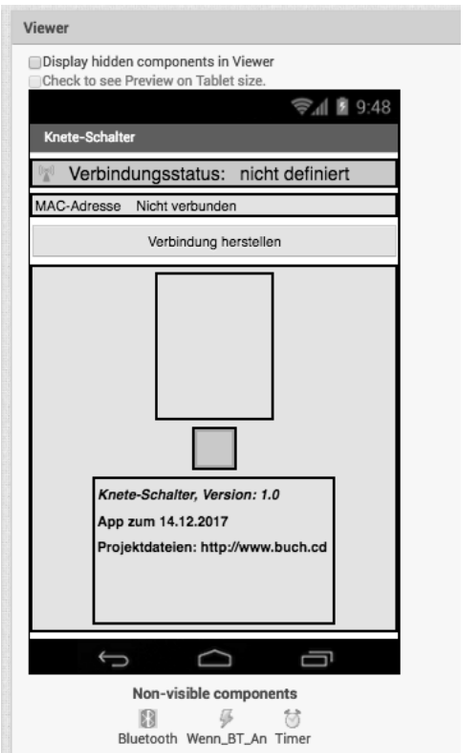
Pour contrôler si un texte a été envoyé par l'interface Bluetooth, il faut vérifier périodiquement si il y a un texte. Ceci est fait dans la minuterie existante.

Configurer les blocs
Les éléments dans la zone **Blocs** dans l'éditeur de bloc peuvent être partiellement configurés. La configuration est exécutée par l'engrenage bleu sur le composant.

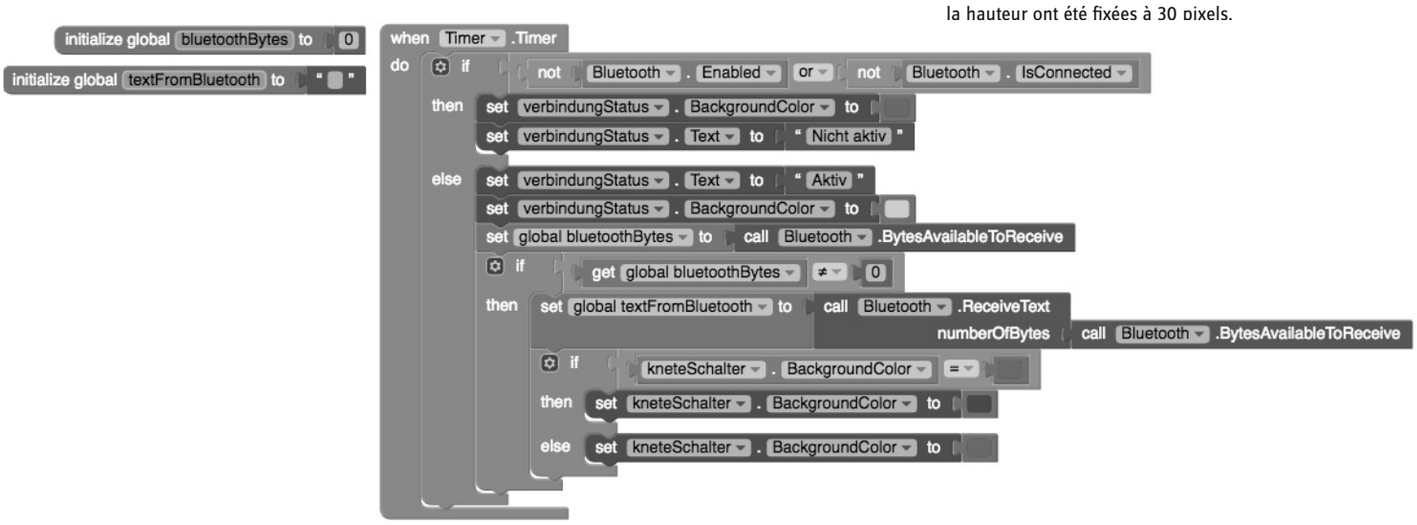


Le bloc if supporte également else et else-if.

Cliquez ensuite sur l 'élément que vous voulez utiliser et le composant se modifie déjà.

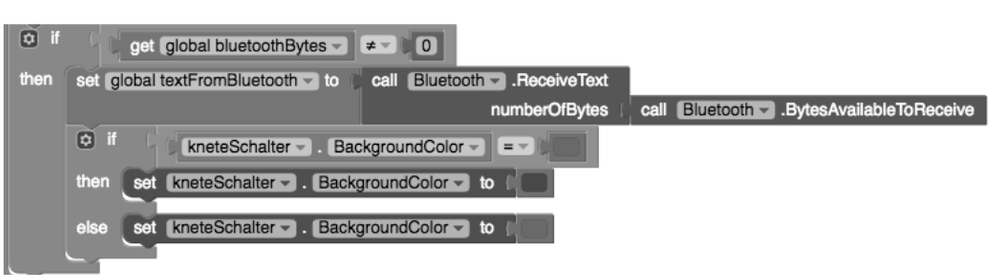


Pour que l'étiquette colorée soit bien visible, la largeur et la hauteur ont été fixées à 30 pixels.



Jusqu'à présent la minuterie n'a été utilisée que pour afficher le statut de la connexion radio.

Le texte est lu par l'interface sans fil via `Bluetooth.ReceiveText`.



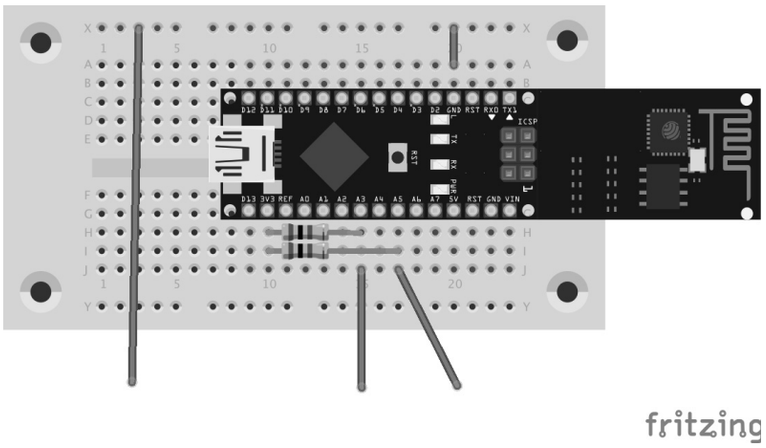
Vous pouvez choisir la couleur à fixer à partir d'un champ de sélection.

15. jour

15e jour

Aujourd'hui dans le calendrier de l'avent

- 1 potentiomètre, 20 MΩ



L'extrémité des trois fils de liaison va dans un morceau de pâte à modeler. Il faut deux résistances à valeur ohmique élevée pour les deux contacts de pâte à modeler.

Contacts de pâte à modeler différentiables

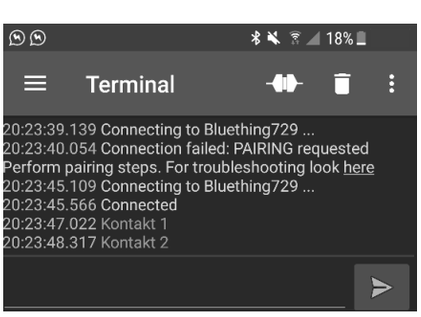
Maintenant utilisez deux contacts de pâte à modeler et indiquez dans une appli quel contact a été enfoncé.

Composants: 1 Breadboard, 3 morceaux de pâte à modeler, 2 résistance 20 MΩ, 4 fils de liaison (différentes longueurs)

Le Sketch

Le programme de ce jour est `Jour15.ino` et se trouve dans le répertoire `Jour15`. Comme pour le jour 14, l'actionnement d'un contact de pâte à modeler est lu via `analogRead`. Il y a une variable supplémentaire pour chaque contact de pâte à modeler. L'ordre est transmis à l'interface sans fil via `HC05.print`.

```
void loop() {
 sensorValue1 = analogRead(sensorPin1);
 sensorValue2 = analogRead(sensorPin2);

 if (sensorValue1 < 256) {
    HC05.print(« contact 1\n »);
  }
  if (sensorValue2 < 256) {
    HC05.print(« contact 2\n »);
  }
  if (sensorValue1 < 256 || sensorValue2 < 256) {
    delay(1000);
  }

}
```



L'actionnement d'un contact de pâte à modeler conduit à la sortie programmée dans le terminal.

L'appli

L'appli **Serial Bluetooth Terminal** est utilisée pour afficher les textes. Connectez-vous d'abord avec la carte IoT et déjà les textes devraient apparaître dès que vous touchez un contact de pâte à modeler.

16e jour

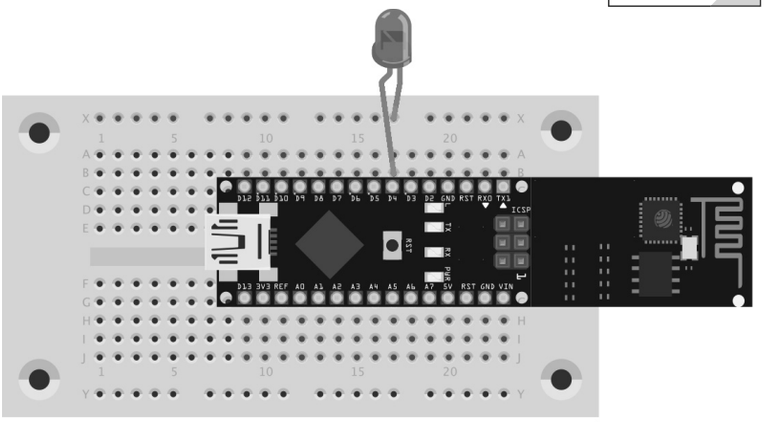
Aujourd'hui dans le calendrier de l'avent

· 1 DEL de clignotement rouge avec résistance en série

Contrôler la DEL de clignotement par l'appli

Activez la DEL de clignotement par un bouton et désactivez-la par un autre bouton.

Composants: 1 Breadboard, 1 DEL de clignotement rouge avec résistance en série



La DEL de clignotement a également un résistance en série et donc aucun autre élément n'est nécessaire pour l'activation.

Le Sketch

Le programme de ce jour est `Jour16.ino` et se trouve dans le répertoire `Jour16`. Le sketch répond à `Marche` et `Arrêt` de l'interface sans fil :

```
if (Text.startsWith(« Marche ») || Text.
startsWith(« MARCHE ») ||  Text.
startsWith(« marche »)){
  marche();
} else if (Text.startsWith(« Arrêt ») || Text.
startsWith(« ARRÊT ») || Text.startsWith(« arrêt »))
{
  arrêt();
}
```

L'activation de la DEL de clignotement se réalise par deux fonctions séparées. Pour cela seule la valeur `LOW` ou `HIGH` via `digitalWrite` est écrite.

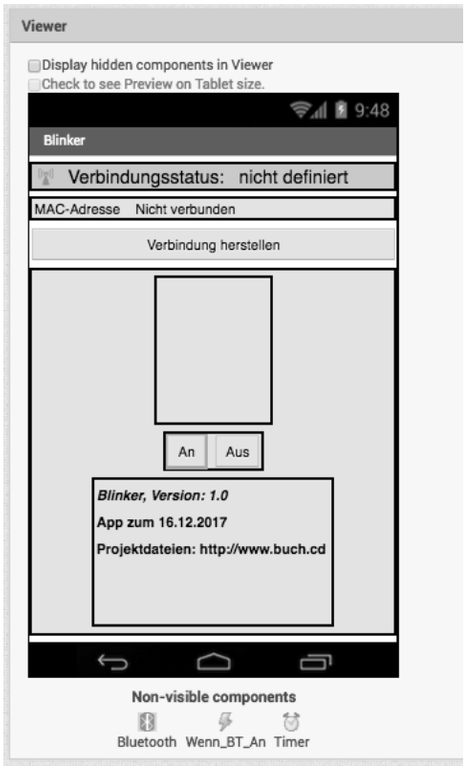
```
void marche() {
  Serial.println(« Marche »);
  digitalWrite(blinkPin, HIGH);

}
void arrêt() {
  Serial.println(« Arrêt »);
  digitalWrite(blinkPin, LOW);

}
```

L'appli

L'appli `Blinker.aia` a deux boutons pour la mise en marche et l'arrêt de la DEL de clignotement.

Pour l'émission on utilise la méthode déjà connue `call Bluetooth.SendText`.

Pensez que pour l'émission le critère final fait également partie de la chaîne.



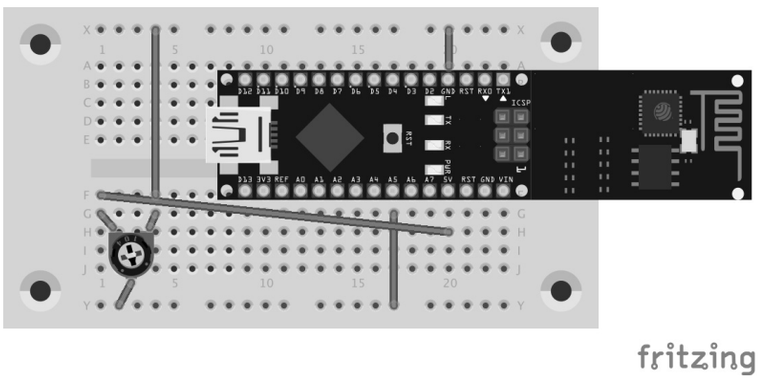
Il y a un **VerticalArrangement** comme élément de positionnement par les deux boutons (Marche et Arrêt).

17. jour

17e jour

Aujourd'hui dans le calendrier de l'avent

- 1 potentiomètre



D'un point de vue programmation, un potentiomètre n'est rien d'autre qu'un capteur analogique.

Affichage des valeurs de résistance

Aujourd'hui vous mesurez la valeur d'un potentiomètre par une entrée analogique et sortez la valeur dans une appli.

Composants: 1 Breadboard, 1 potentiomètre 25 kΩ, 4 fils de liaison (différentes longueurs)

Le Sketch

Le programme de ce jour est `Jour17.ino` et se trouve dans le répertoire `Jour17`. Les deux variables `sensorPin` et `sensorValue` sont d'abord définies. Le potentiomètre fait partie de l'entrée analogique 5 et par conséquent `sensorPin` est fixé sur `A5`.

```
int sensorPin = A5;
int sensorValue = 0;
```

Dans la méthode `loop` la valeur est lue par `analogRead` et transmise à l'interface sans fil. Ensuite il y a une pause de 2 000 ms et la valeur est de nouveau lue :

```
void loop() {
 sensorValue = analogRead(sensorPin);
 HC05.print(« valeur potentiomètre  = »);
 HC05.println(sensorValue);
 delay(2000);
}
```

L'appli

L'appli **Serial Bluetooth Terminal** est utilisée pour afficher les textes. Connectez-vous d'abord à la carte IoT et déjà vous voyez les valeurs du potentiomètre.

18e jour

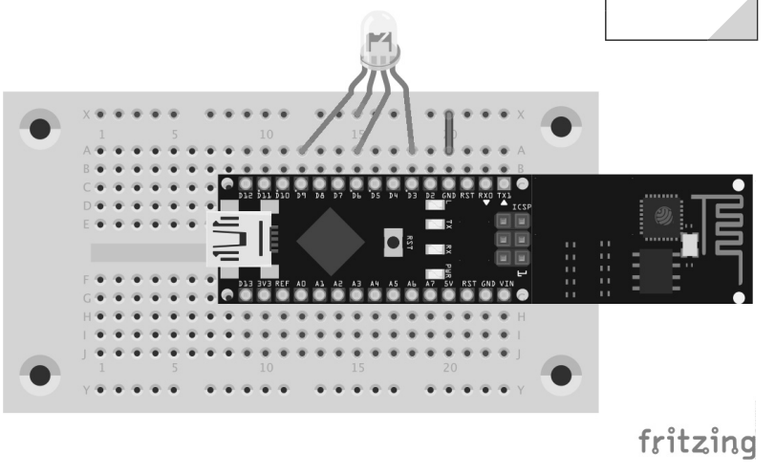
Aujourd'hui dans le calendrier de l'avent

· 1 câble jumper

Chenillard RVB

Vous vontrôlez la durée de clignotement d'une DEL RVB par une appli. La DEL clignote dans des couleurs en alternance. Vous pouvez également arrêter le clignotement.

Composants: 1 Breadboard, 1 DEL RVJ avec résistance en série, 1 câble jumper



Le circuit est très compact par les résistances intégrées.

Le Sketch

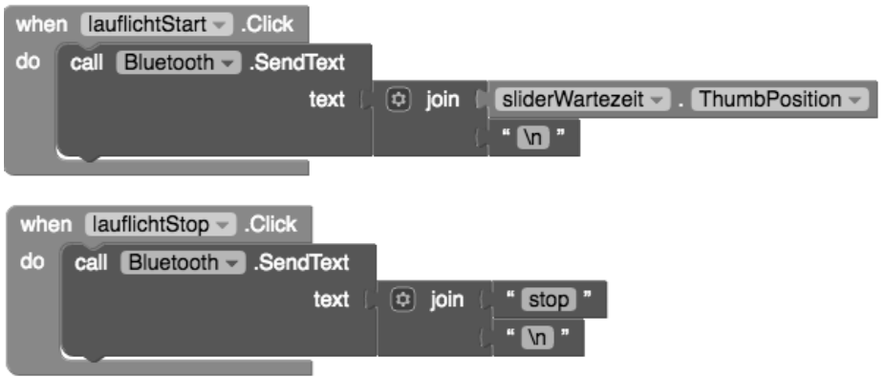
Le programme de ce jour est `Jour18.ino` et se trouve dans le répertoire `Jour18`. Le sketch répond au texte `stop` et sur un nombre envoyé. Dans le cas de `stop` la DEL RVB est éteinte. S'il y a un nombre, une des trois DELS de la DEL RVB est utiliser en pause entre l'enclenchement et l'arrêt.

```
if (caractère == ,\n‘) {
 Serial.println(Text);
 if (Text.startsWith(“stop”)){
  arrêt();
  Durée de clignotement = 0
 } else {
  Durée de clignotement = durée de clignotement lue(texte)
 }
```

L'appli

L'appli `Chenillard_RVB.aia` a un slider et deux boutons pour enclencher et arrêter le clignotement.

Le temps d'attente est envoyée par l'ordre déjà connu `Bluetooth.SendText` ou le texte `stop` par l'interface sans fil.



Le temps de pause n'est pas spécialement codé, seule la valeur du slider est envoyée suivie d'un \n final.

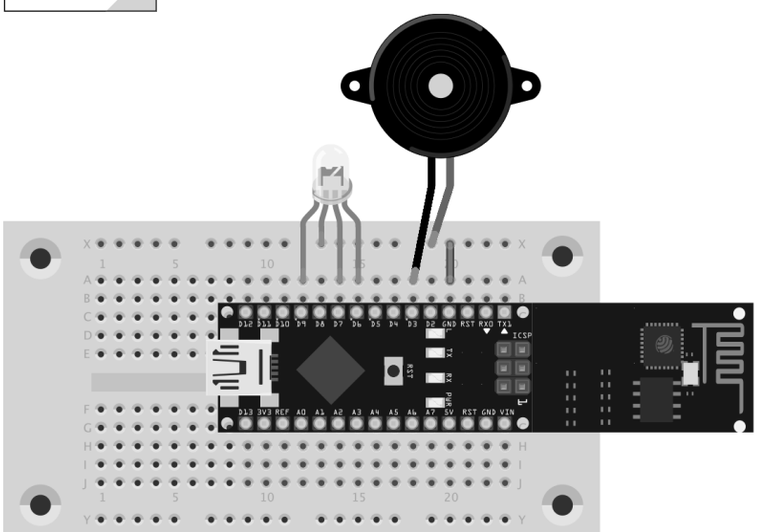
La valeur minimale du slider est 0 et la valeur maximale est 2 000. Les valeurs pour le temps de pause sont en millisecondes.

19. jour

19e jour

Aujourd'hui dans le calendrier de l'avent

- 1 câble jumper



La DEL RVB n'a pas besoin de résitances en série. Vous utilisez le câble jumper pour établir la connexion à la terre.

Cette appli utilise les éléments déjà connus des autres applis pour l'établissement de la connexion. Vous établissez la connexion via **Établir la connexion**.

Appli pour sélection des applis hardware.

Aujourd'hui deux fonctionnalités seront combinées dans un circuit : Reproduction sonore avec piezo et clignotement d'une DEL RVB. Vous sélectionnez les fonctionnalités correspondantes par l'appli.

Composants: 1 Breadboard, 1 DEL RVJ avec résistance en série, 1 Piezo, 1 câble jumper

Le Sketch

Le programme de ce jour est `Jour19.ino` et se trouve dans le répertoire `Jour19`. Les broches correctes doivent d'abord être définies comme variables :

```
//---- Broches de la DEL RVB
int redPin = 9;
int greenPin = 7;
int bluePin = 8;

//---- Pin Piezo
int piezo = 3;
```

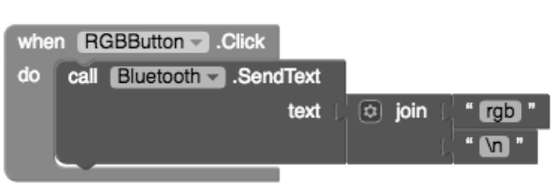
Le sketch répond aux ordres `piezo` et `rgb`. Les méthodes ont été adoptées à partir des sketches précédents :

```
if (Text.startsWith(« piezo »)){
 arrêt();
 playMelody();
} else if (Text.startsWith(« rvb »)) {
 clignoter(500);
};
```

L'appli

L'appli `App_changeur.aia` a deux boutons pour changer entre Piezo et RVB

La fonctionnalité est représentée en deux blocs when.click. Elle n'est adaptée qu'au texte à envoyer.



Les blocs sont très similaires. Dans de tels cas, assurez-vous d'avoir sélectionné le bouton correct dans le bloc when.click.

20e jour

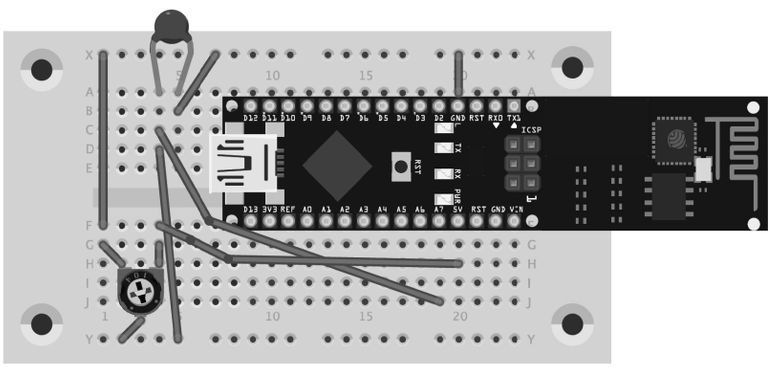
Aujourd'hui dans le calendrier de l'avent

· 1 sonde NTC

Capteur thermique dans l'appli

Dans le projet d'aujourd'hui une sonde NTC est utilisé pour mesurer la température. Vous pouvez tester le projet, par exemple en approchant une bougie avec la sonde NTC sur la couronne de l'Avent.

Composants: 1 Breadboard, 1 sonde NTC, 1 potentiomètre 15 kΩ, 6 fils de liaison (différentes longueurs)



N'oubliez pas d'utiliser la résistance 10 kΩ pour ne pas être insatisfait des résultats de mesure.

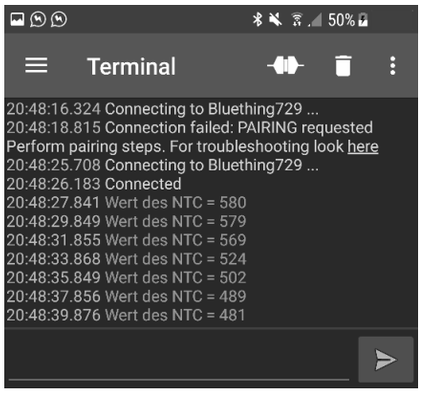
Le Sketch

Le programme de ce jour est `Jour20.ino` et se trouve dans le répertoire `Jour20`. Le sketch est très simple à mettre en place. Après l'appel de `analogRead` la valeur est envoyée sur l'appli via le Bluetooth, ensuite il y a un temps d'attente de 2 secondes et la valeur est de nouveau lue et envoyée :

```
void loop() {
 sensorValue = analogRead(sensorPin);
 HC05.print(« valeur du NTC = »);
 HC05.println(sensorValue);
 delay(2000);
}
```

L'appli

L'appli **Serial Bluetooth Terminal** est utilisée pour afficher les textes. Connectez-vous d'abord à la carte IoT et déjà vous voyez les valeurs de la sonde NTC. Si les valeurs sont plus petites, c'est un signe d'augmentation de la température.



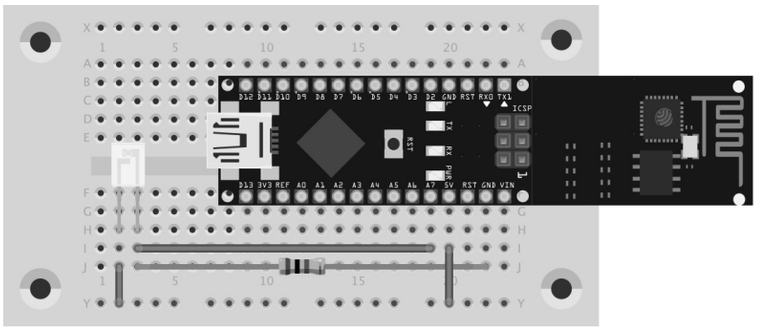
Comme pour les autres projets les données reçues sont représentées en vert. N'oubliez pas la requête **delay** dans le sketch pour ne pas avoir trop de valeurs.

21. jour

21e jour

Aujourd'hui dans le calendrier de l'avent

- 1 phototransistor
- 1 résistance 1 kΩ



La mise en place est similaire à celle du Jour 20. Attention : Une autre résistance est utilisée et la connexion au VCC (+5 V) est importante.

Mesure de la luminosité et de l'obscurité dans l'appli. Maintenant vous disposez d'un autre capteur pour vos projets. Vous pouvez afficher la luminosité et l'obscurité avec le phototransistor. C'est ce que vous allez apprendre aujourd'hui. En sachant cela, vous pouvez créer rapidement une barrière lumineuse et ainsi avoir un système d'alarme avec affichage dans une application.

Composants: 1 Breadboard, 1 phototransistor, 1 résistance 1 kΩ, 3 fils de liaison (différentes longueurs)

Le Sketch

Le programme de ce jour est `Jour21.ino` et se trouve dans le répertoire `Jour21`. Le sketch ressemble à celui du jour précédent. Si vous voulez voir plus de valeurs sur le Smartphone, vous devez diminuer la durée de pause lors de la requête `delay`.

```
void loop() {
 sensorValue = analogRead(sensorPin);
 HC05.print(« valeur du phototransistor = »);
 HC05.println(sensorValue);
 delay(2000);
}
```

L'appli

L'appli **Serial Bluetooth Terminal** est utilisée pour afficher les textes. Connectez-vous d'abord à la carte IoT et déjà vous voyez les valeurs de la sonde NTC. Regardez comment les valeurs changent avec la lumière et l'obscurité.

22e jour

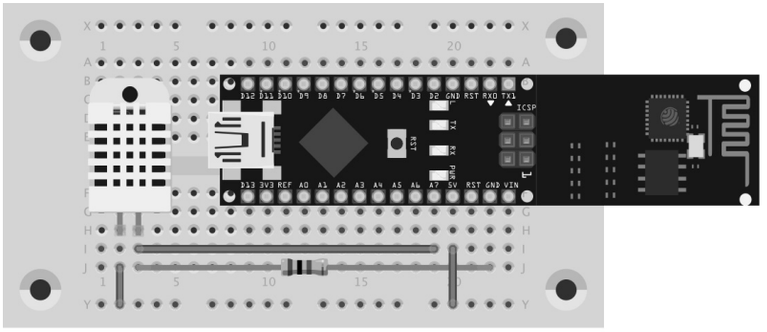
Aujourd'hui dans le calendrier de l'avent

- 1 capteur d'humidité
- 1 résistance 1 kΩ

Humidimètre

Aujourd'hui vous avez de nouveau un capteur, un capteur d'humidité. Vous pouvez ainsi mesurer l'humidité.

Composants: 1 Breadboard, 1 capteur d'humidité, 1 résistance 1 kΩ, 3 fils de liaison (différentes longueurs)



Le circuit ressemble au circuit du jour 21. Seul le capteur doit être remplacé.

Le Sketch

Le programme de ce jour est `Jour22.ino` et se trouve dans le répertoire `Jour22`. Comme pour les jours précédents le capteur est lu par `analogRead` et la valeur est transmise à l'interface :

```
void loop() {
 sensorValue = analogRead(sensorPin);
 HC05.print(« valeur du capteur d'humidité = »);
 HC05.println(sensorValue);
 delay(2000);
}
```

L'appli

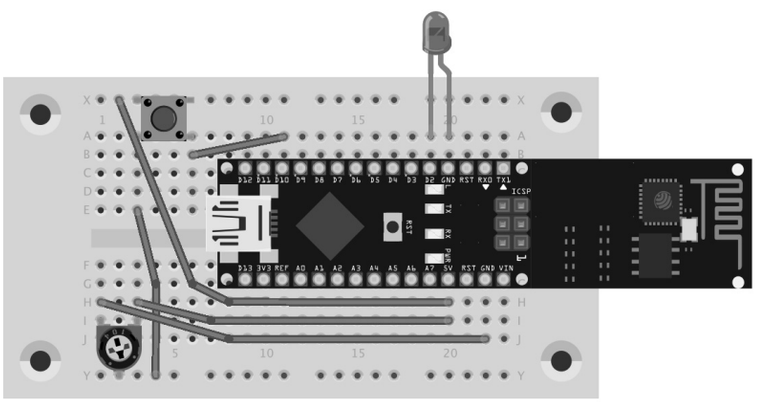
L'appli **Serial Bluetooth Terminal** est utilisée pour afficher les textes. Connectez-vous d'abord à la carte IoT et déjà vous voyez les valeurs du capteur d'humidité. Prenez le capteur d'humidité entre deux doigts et observez l'évolution des valeurs.

23. jour

23e jour

Aujourd'hui dans le calendrier de l'avent

- 1 câble jumper



Pour que le bouton-poussoir fonctionne également correctement, un diviseur de tension est mis en place par le potentiomètre. Vous pouvez aussi utiliser le câble jumper à la place des fils de liaison.

Décodeur

Le projet d'aujourd'hui est un petit jeu : Paramétrez un code dans une appli. Transmettez ce code à la carte IoT. Maintenant le code doit être saisi avec le bouton-poussoir. Lorsque le code a été correctement saisi, la DEL rouge s'allume et la saisie correcte du code est indiquée dans l'appli. Sinon la DEL rouge ne s'allume pas et la saisie incorrecte du code est indiquée dans l'appli.

Composants: 1 Breadboard, 1 bouton-poussoir, 1 DEL rouge avec résistance en série, 1 potentiomètre 15 kΩ, 5 fils de liaison (différentes longueurs)

Le Sketch

Le programme de ce jour est `Jour23.ino` et se trouve dans le répertoire `Jour23`. Le code est transmis sous la forme CNNN :

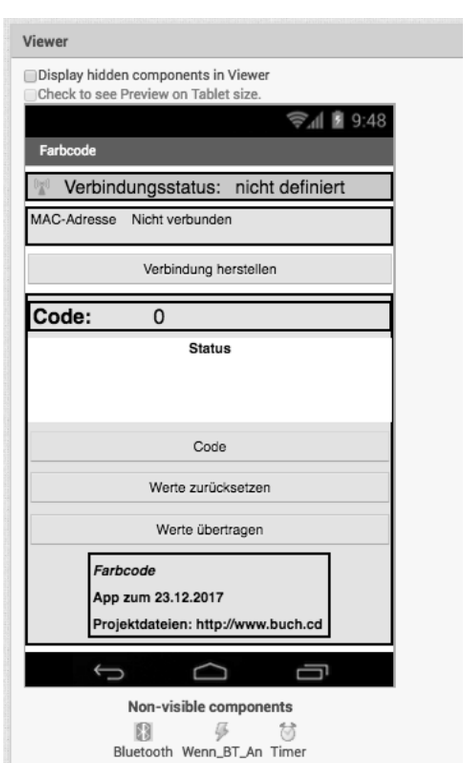
```
if (Text.startsWith("C")){
      nombretouches = Text.substring(Text.
indexOf("C")+1).toInt();
```

Le joueur a cinq secondes en tout pour saisir le code :

```
interval = millis() - start;
Serial.println(interval);
while (interval < 5000) {
  pressed += lecturepalpeur();
  delay(100);
  interval = millis() - start;
}
```

On vérifie si une touche a été appuyée par la méthode `lecturepalpeur`. On attentd 100 ms entre les contrôles. Le résultat est restitué sur l'appli :

```
if (pressed == nombretouches) {
  HC05.print(« OUI »);
  digitalWrite(redPin, HIGH);
  delay(5000);
 digitalWrite(redPin, LOW);
} else {
  HC05.print(« NON »);
}
```



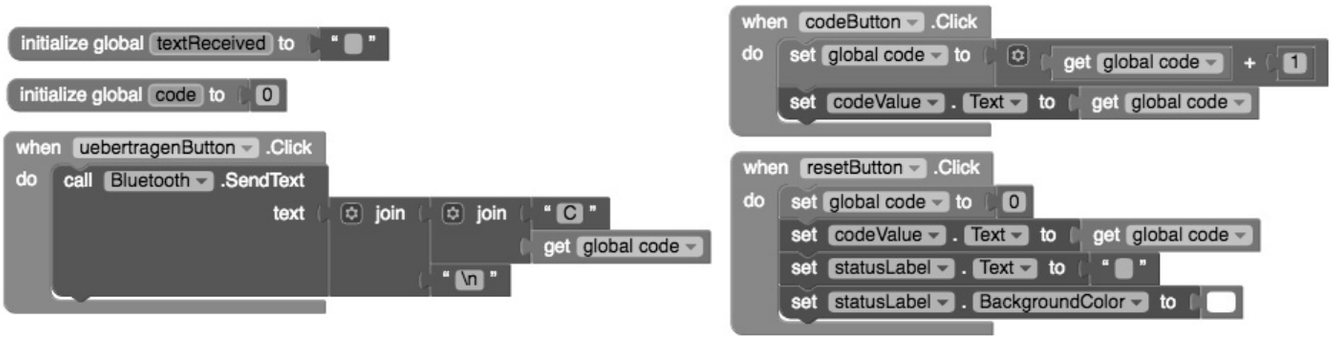
Une étiquette de couleur indique si le code a été correctement saisi ou non par le palpeur matériel.

L'appli

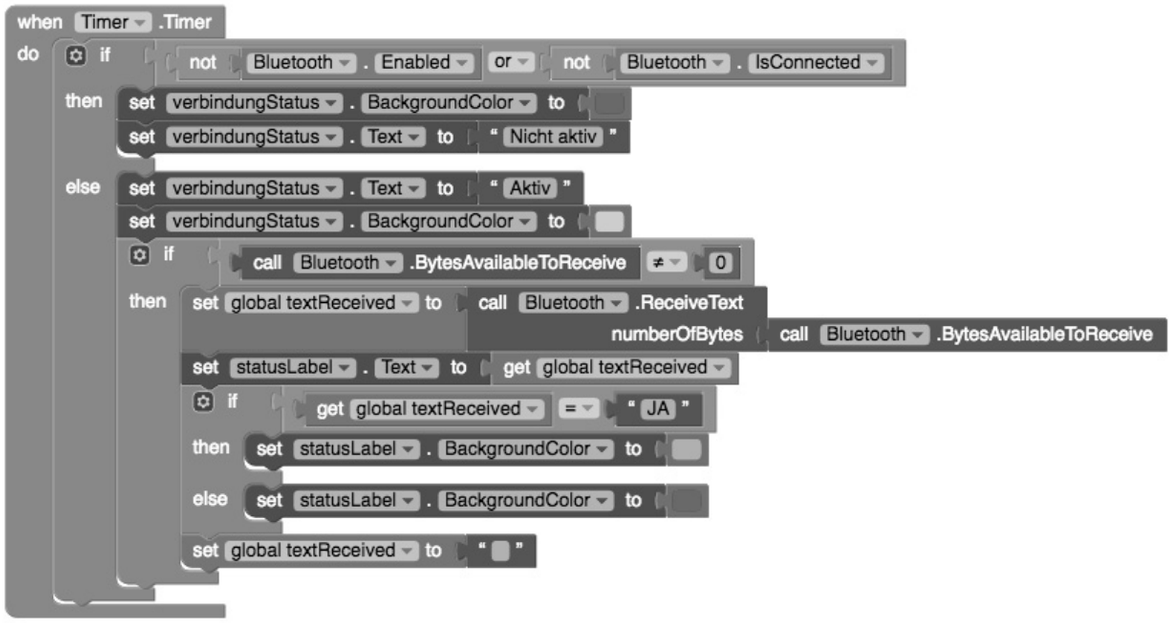
L'appli `codecouleur.aia` a trois boutons et deux étiquettes. Le code saisie est affiché avec une des étiquettes et le statut du jeu avec la deuxième étiquette. Si le code a été correctement saisi par le palpeur matériel alors la couleur de fond de l'étiquette pour le statut du jeu est verte, sinon elle est rouge.

Le code est stocké dans une mémoire tampon dans une variable globale et transmis ensuite avec `Bluetooth.SendText`.

La réception des textes du bloc IoT a lieu dans la minuterie périodique. Pour cela la minuterie, qui a déjà servi à l'affichage du statut de la connexion, est utilisée.



Étant donné que 3 boutons sont nécessaires pour le jeu, il y a également trois blocs when.click.



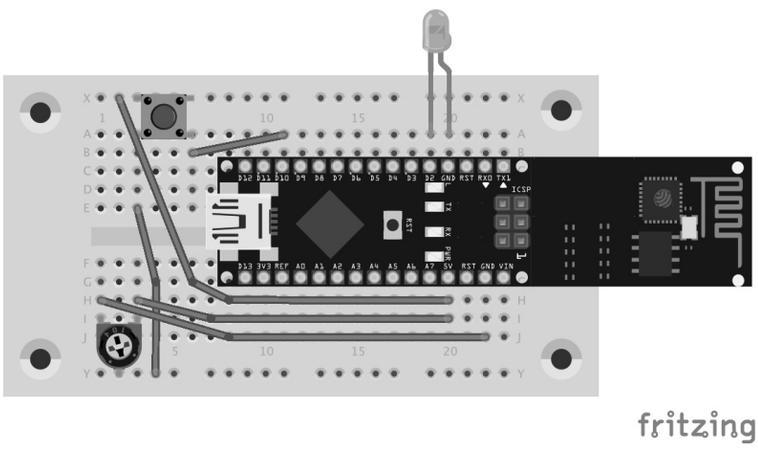
À réception de OUI la couleur de fond est verte, autrement elle rouge.

24. jour

24e jour

Aujourd'hui dans le calendrier de l'avent

- 1 poussoir



Le circuit d'aujourd'hui est semblable au circuit du jour précédent. Seule la DEL rouge est remplacée par une DEL verte.

Jeu de réaction

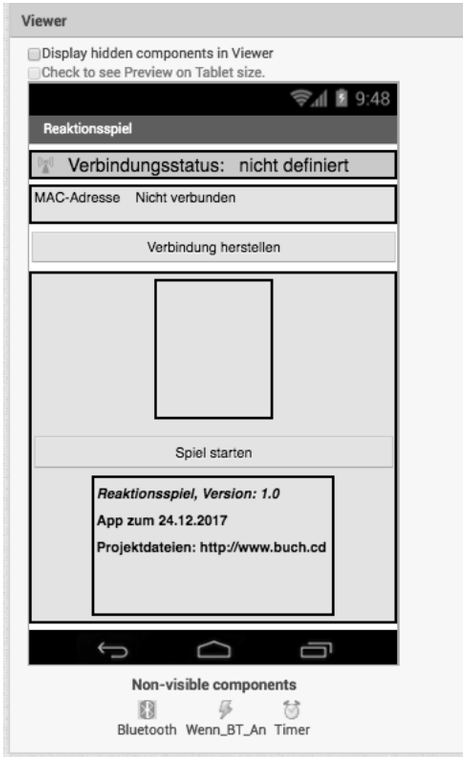
Pour conclure le calendrier de l'Avent il y a encore un petit jeu de réaction. Vous démarrez le jeu par un bouton dans l'appli. La DEL sur le Breadboard s'allume et dès que la DEL s 'éteint vous devez appuyer sur le palpeur. La durée est affichée sur l'appli.

Composants: 1 Breadboard, 1 bouton-poussoir, 1 DEL verte avec résistance en série, 1 potentiomètre 15 kΩ, 5 fils de liaison (différentes longueurs)

Le Sketch

Le programme de ce jour est `Jour24.ino` et se trouve dans le répertoire `Jour24`. La durée est mesurée via la fonction `millis` jusqu'à ce que le palpeur soit appuyé :

```
if (Text.startsWith(“START”)){

   digitalWrite(greenPin, HIGH);
   delay(1000);
   digitalWrite(greenPin, LOW);

   //temps de démarrage
   int durée = millis();

   Serial.println(interval);
   while (lecturepalpeur() != 1) {
     delay(100);
   }
   durée =millis() - durée;
   HC05.print(durée);
   digitalWrite(greenPin, LOW);
 }
```

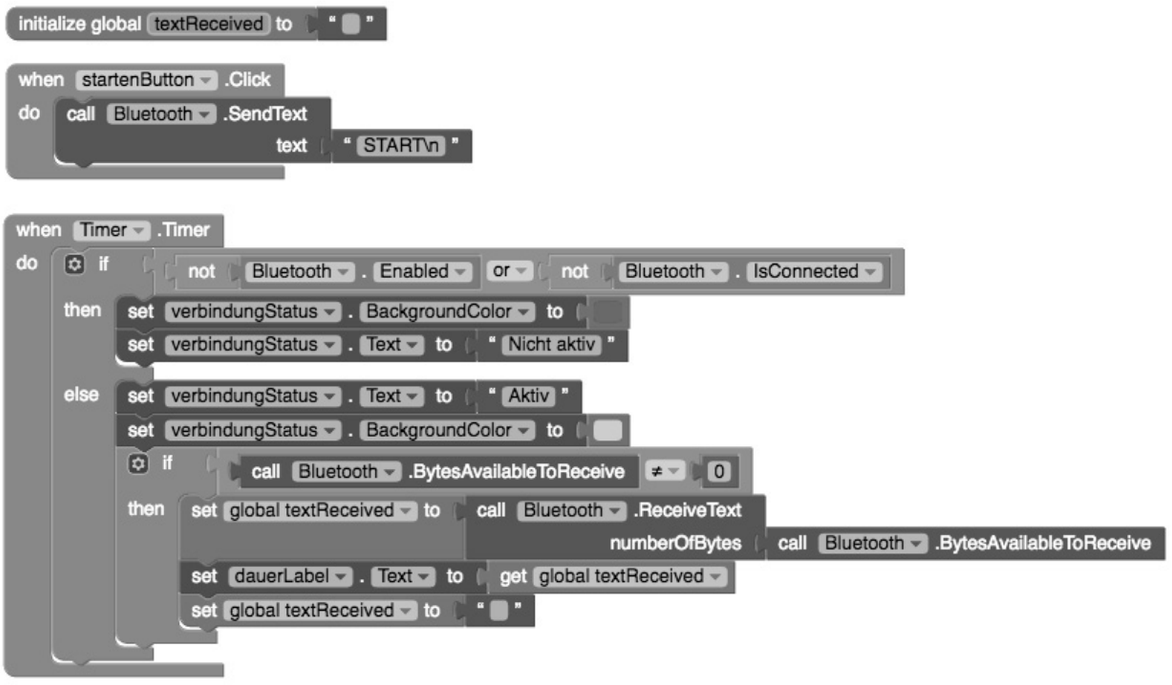


Sur le bouton **Démarrer le jeu** se trouve une étiquette sans texte. Ce n'est que lorsque un texte arrive de la carte IoT que le le texte est placé sur l'étiquette.

L'appli

L'appli `jeu de réaction.aia` a un bouton avec la désignation **Démarrer le jeu**. En actionnant le bouton, le texte est envoyé par l'interface sans fil. Le texte qui provient de la carte IoT est indiqué dans l'étiquette sur le bouton.

Pour le démarrage du jeu, le bloc `when startButton.click` est analysé en cliquant sur le bouton **Démarrer le jeu** et le texte DÉMARRER est transmis par l'interface sans fil.



La lecture de l'interface sans fil devrait toujours se produire dans une minuterie. Car on ne sait pas précisément quand un paquet de données arrive.

Joyeux Noël !