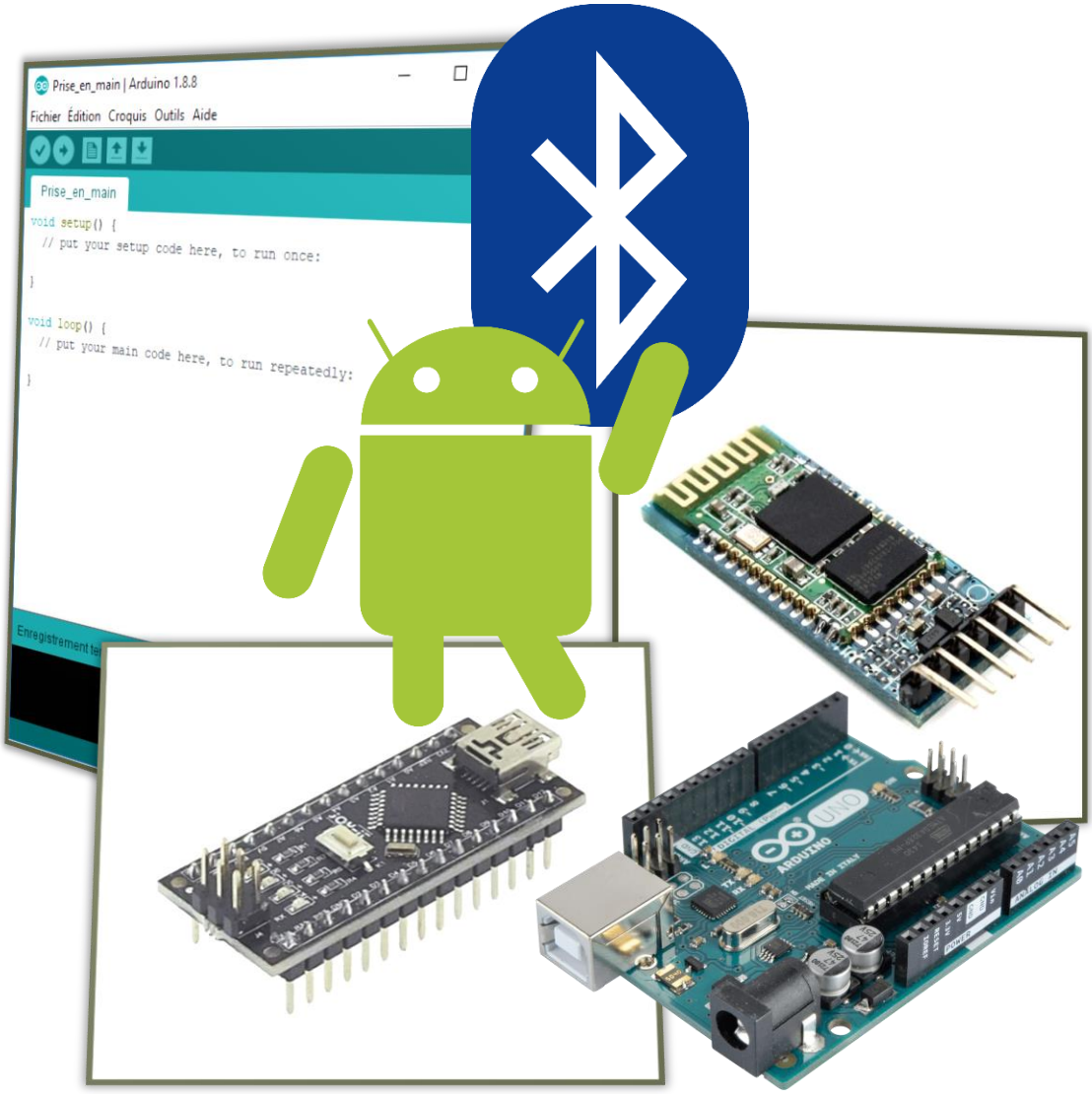
BLUETOOTH

TUTORIEL



LYCEE CLOS MAIRE

TABLE DES MATIERES

INTRODUCTION

OBJECTIFS

Le but de ce document est de vous faire découvrir la communication Bluetooth entre une carte Arduino et une application Android.

CONSIGNES

Lire ce tutoriel dans l'ordre et en entier.

A chaque encadré comme celui-ci se trouve une manipulation à effectuer ou une question à répondre.

Vous répondrez sur un document Word soigneusement présenté et intitulé

« **NOM_Arduino_Bluetooth** ».

Commencez par créer votre document Word de réponse et enregistrez-le dans votre espace personnel.

MATERIELS NECESSAIRES

Deux possibilités : soit vous travaillez avec la carte Nano soit vous travaillez avec la carte UNO.

Carte Nano :

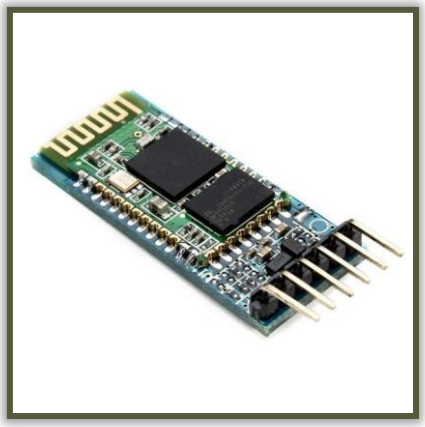
- Carte Nano
- Câble USB-Mini-B
- Plaque de montage
- Module Bluetooth HC-05
- Smartphone Android

Carte UNO :

- Carte UNO
- Câble USB-B
- Module Bluetooth HC-05
- Smartphone Android

LE MODULE BLUETOOTH HC-05

Le module Bluetooth HC05 permet d'ajouter une liaison Bluetooth à la carte Arduino et ainsi de communiquer avec d'autres interfaces dotées de la technologie Bluetooth.



APP INVENTOR

App Inventor pour Android est une application développée par Google. Elle est actuellement entretenue par le Massachusetts Institute of Technology (MIT).

Elle permet de simplifier le développement des applications sous Android et le rend accessible pour un large public puisqu'elle est basée sur une interface graphique similaire à Scratch.

PREMIER PROGRAMME

Le but de ce premier programme est d'envoyer une information binaire (1 ou 0) via un bouton sur une application Android afin de commander la LED présente sur la carte Arduino.

DECOUVERTE APP INVENTOR

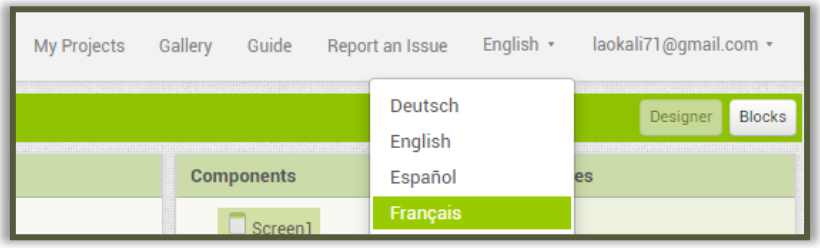
Aller sur le site de APP Inventor : http://appinventor.mit.edu/

Cliquer en haut de la page sur « **Create Apps** ! »

Se connecter avec **une adresse Gmail**.

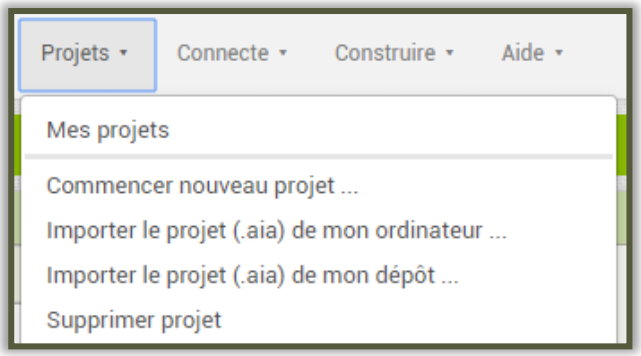
Vous arrivez alors à l'interface de l'application.

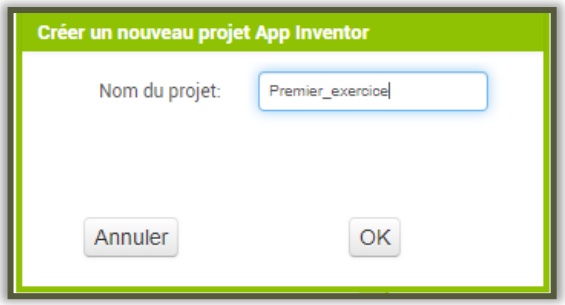
En haut à droite de l'écran changer la langue pour mettre l'interface en **Français**.



Créer un nouveau projet en cliquant en haut à gauche sur « Projets » puis « Commencer un nouveau projet ».

Nommer ce projet « **Premier_exercice** ».

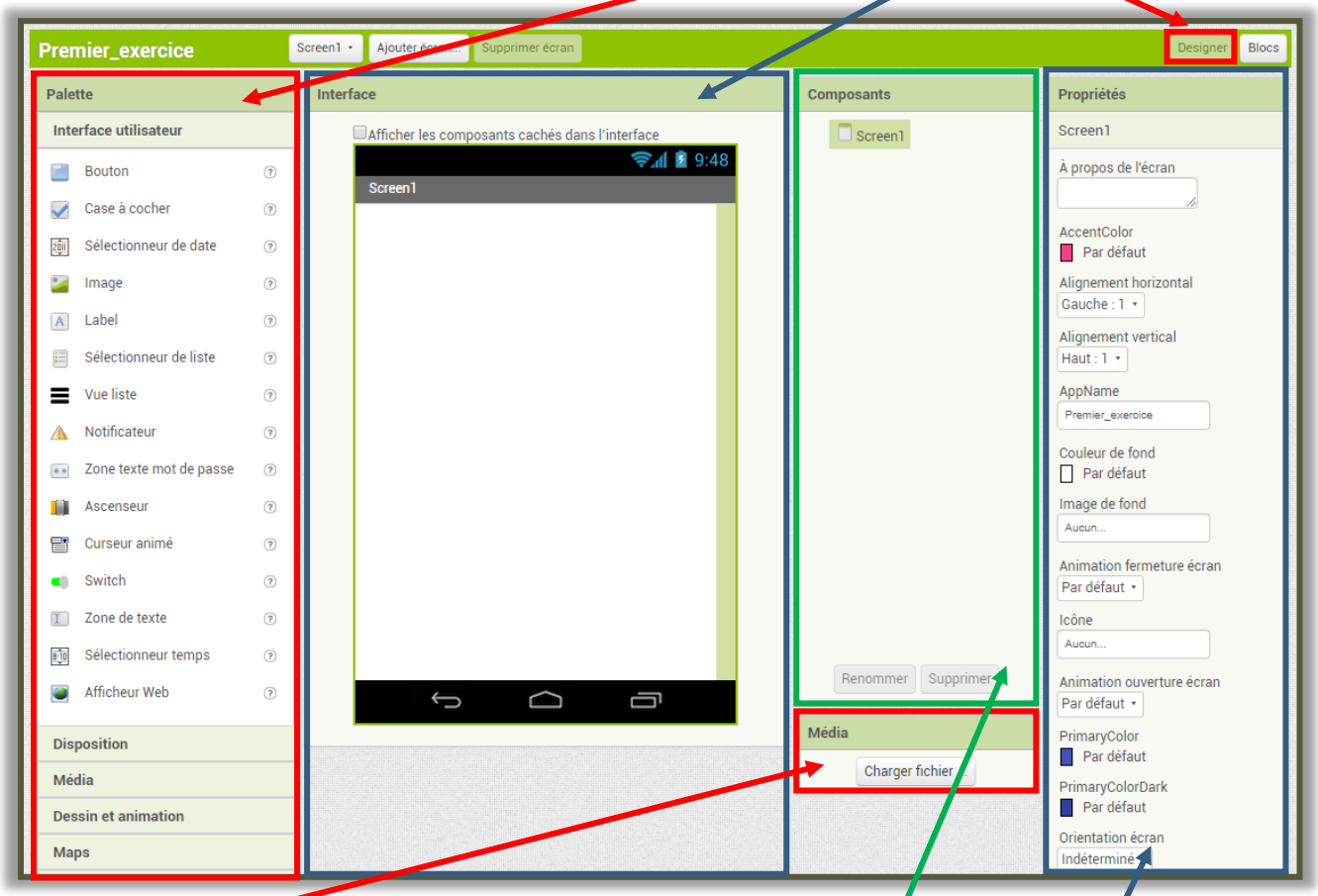




Avant de commencer à designer et à coder votre application, quelques explications sur l'interface.

Cette première interface où vous vous trouvez est l'interface « Designer », elle permet de placer tous les éléments qui vont être visibles sur l'application par l'utilisateur.

Il suffit alors de glisser-déposer les éléments depuis la « palette » vers « l'interface ».



Chaque élément placé est alors ensuite listé dans la partie « Composants » et possède des caractéristiques modifiables dans la partie « Propriétés ».

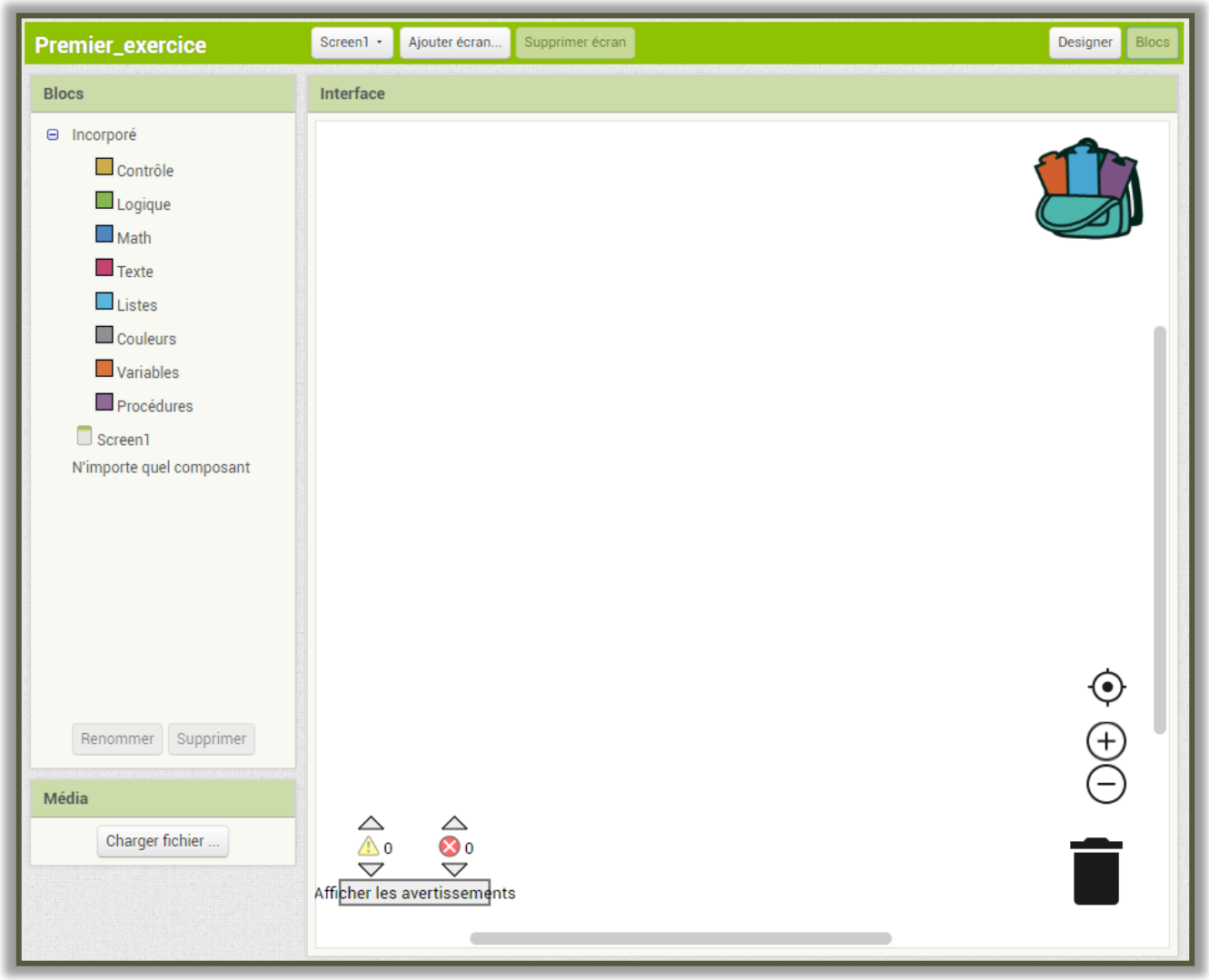
La partie « Média » permettra d'ajouter des fichiers médias à l'application tel que des sons, des images, des vidéos, etc.

Il est possible de créer des applications avec plusieurs écrans différents. Il est alors possible de gérer les différents les écrans via la barre supérieure de l'interface.

La partie suivante concerne la programmation de l'application. Une fois que les divers composants sont placés (boutons, afficheur de texte, etc.) il faut alors les programmer de façon à effectuer des actions.

Pour accéder à l'interface de programmation de l'écran 1 il faut cliquer sur le bouton « Blocs ».

Cliquer sur le bouton « Blocs ».



Pour programmer l'application il suffit de prendre des blocs dans la partie « Blocs » et des les glisser-déposer dans l'interface.

Le sac à dos en haut à droite permet de stocker ou de récupérer des parties de blocs déjà stockés. Pour stocker des éléments il suffit de les glisser déposer sur le sac à dos.

PROGRAMMATION APP INVENTOR

La première partie de la programmation consiste à créer un bouton permettant de se connecter au module Bluetooth de l'Arduino et ainsi d'accéder au deuxième écran de l'application qui sera l'écran de contrôle.

Cliquer sur « Ajouter écran » puis « ok ».

Il y a maintenant deux écrans sur l'application.

Sur le Screen, ajouter un « Bouton » sur l'application.

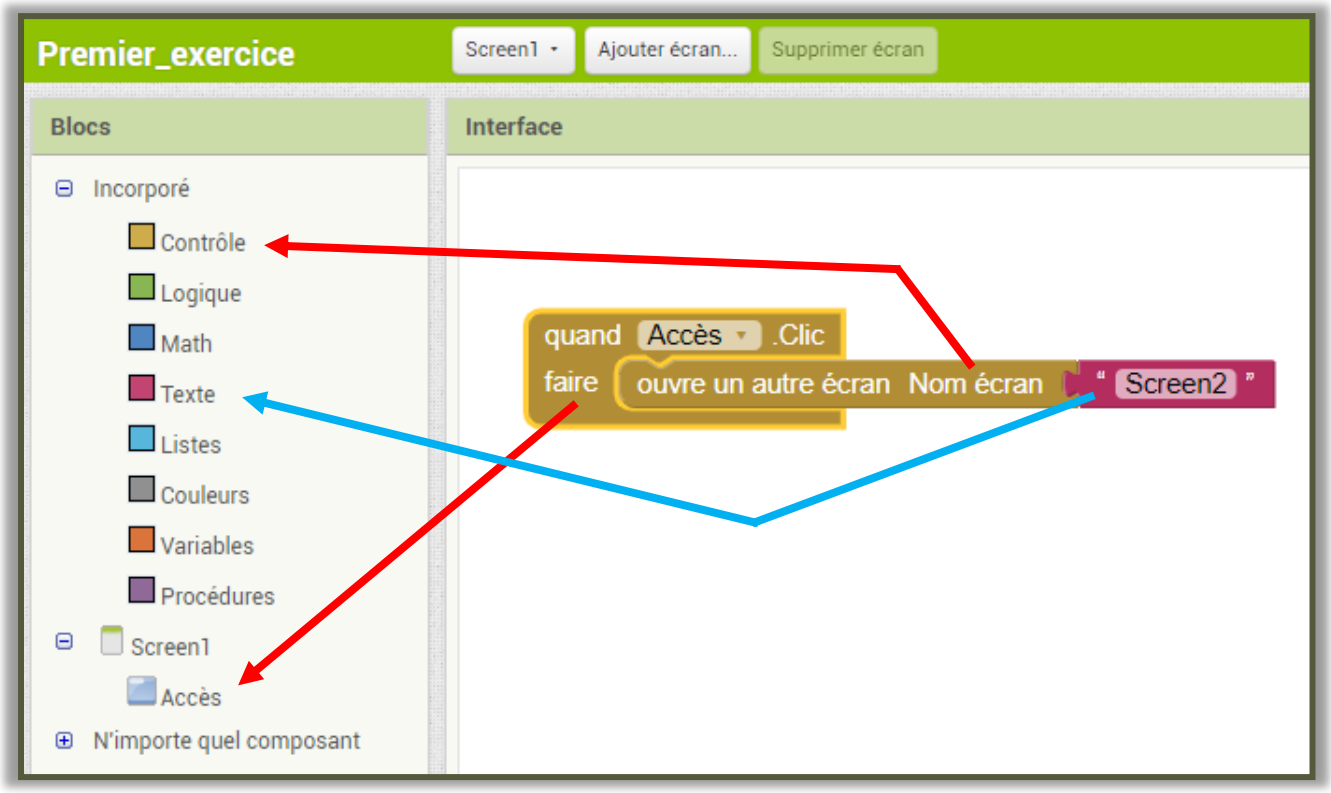
Dans le menu composants, renommer le nom du bouton par « Accès ».

Dans le menu Propriétés, mettre le texte « Accéder à l'écran de contrôle » puis cocher « Remplir parent » pour la largeur du bouton.

Vous devez obtenir le résultat comme ci-dessous.

Accéder maintenant à l'interface de programmation en cliquant sur « Blocs » en haut à droite de l'écran.

Voici le programme à réaliser pour cet écran :



Reproduire le programme ci-dessus.

Passer maintenant sur le Screen2.

Il suffit d'aller chercher chaque bloc dans le menu « Blocs » et de les glisser-déposer dans l'interface.

Dans le menu « Palette », cliquer sur l'onglet « Connectivité » et ajouter un « Client Bluetooth » à l'application.

Dans l'onglet « Capteurs », ajouter une « Horloge » à l'application.

Le client Bluetooth permettra d'utiliser le Bluetooth du smartphone sur lequel l'application est installée.

L'horloge permettra de définir des temps pour lesquels les informations seront envoyer par Bluetooth.

Ajouter un « Sélectionneur de liste ».

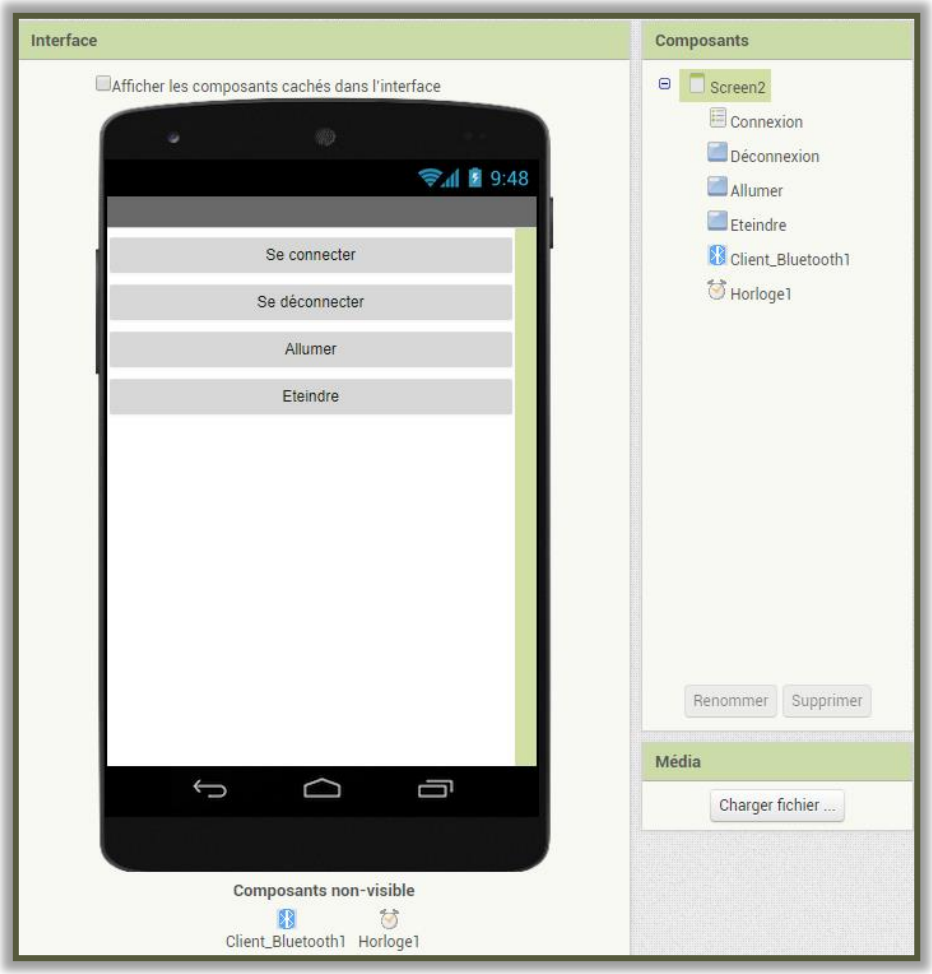
Dans le menu composants, renommer le nom du sélectionneur par « Connexion ».

Dans le menu Propriétés, mettre le texte « Se connecter » puis cocher « Remplir parent » pour la largeur du sélectionneur.

Ajouter 3 boutons.

Dans le menu composants, renommer le nom des boutons par « Déconnexion », « Allumer » et « Eteindre ».

Dans le menu Propriétés, pour chaque bouton, mettre le texte « Se déconnecter », « Allumer », « Eteindre » puis cocher « Remplir parent » pour la largeur des boutons.



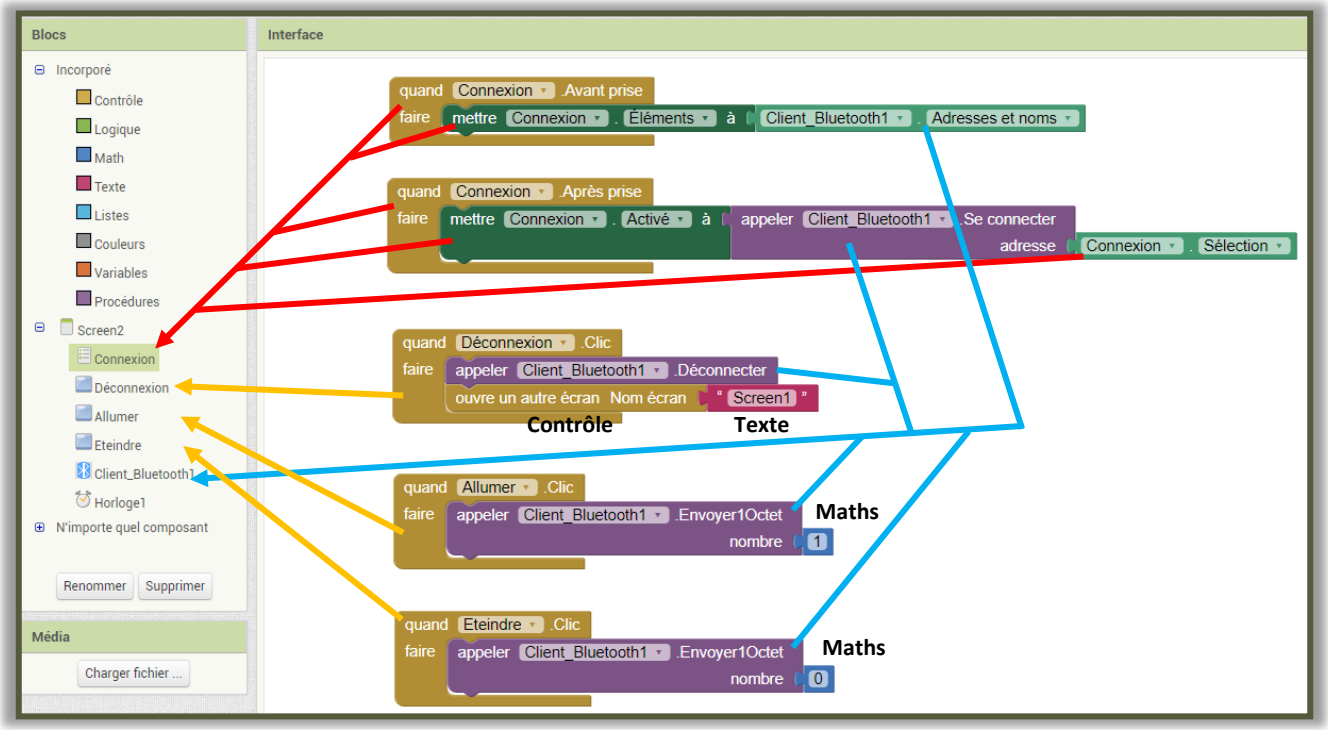
Accéder maintenant à l'interface de programmation en cliquant sur « Blocs » en haut à droite de l'écran.

Vous devez arriver au même design que ci-dessous.

Voici le programme à réaliser pour cet écran :

Explication du programme :

- Le sélecteur de liste « Connexion » a en liste les différents client Bluetooth déjà appairé au



smartphone.

- Lorsque l'on sélectionne un client Bluetooth dans la liste le smartphone se connecte à celui-ci.
- Lorsqu'on clique sur le bouton « Déconnexion », cela déconnecte le client Bluetooth et l'application reviens à l'écran 1.
- Lorsqu'on clique sur le bouton « Allumer » on envoie un octet codant le chiffre 1 sur la liaison Bluetooth.
- Lorsqu'on clique sur le bouton « Eteindre » on envoie un octet codant le chiffre 0 sur la liaison Bluetooth.

Le but sera donc de récupérer ces valeurs d'octet envoyées en Bluetooth pour les traiter avec l'Arduino. Si on reçoit l'octet codant le chiffre 1 alors on allume la LED, si on reçoit l'octet codant le chiffre 0 alors on éteint la LED.

Reproduire le programme ci-dessus.

TESTER L'APPLICATION

Il est possible de tester l'application suivant plusieurs méthodes.

Vous trouverez toutes ces méthodes dans le fichier « Tutoriel_Test_APP_Inventor »

Tester l'application.

PROGRAMMATION ARDUINO

Il est maintenant nécessaire créer un programme Arduino en adéquation avec l'application Android pour recevoir et traiter les informations correctement.

Ouvrir le logiciel Arduino.

Enregistrer sous le programme dans votre espace personnel en le nommant

« **NOM_Arduino_Bluetooth** »

Pour utiliser la liaison série de la carte Arduino avec le Bluetooth il est nécessaire d'utiliser une bibliothèque complémentaire : la bibliothèque Software Serial.

Avant le setup du programme, entrer les lignes suivantes :

```
#include <SoftwareSerial.h>
SoftwareSerial BT(10, 11);
```

La première ligne introduit la bibliothèque.

La deuxième ligne permet de définir la liaison Bluetooth comme s'appelant **BT** et de définir les bornes où les broches TX et RX du module Bluetooth sont branchées sur l'Arduino. TX sur D10 et RX sur D11.

Effectuer les câblages des broches TX et RX comme définit ci-dessus et brancher les broches VCC et GND respectivement sur les bornes 5V et GND.

Pour recevoir des informations venant du module Bluetooth nous allons créer une variable prenant comme valeurs les données reçues par le module. Cette variable sera de type « char » c'est-à-dire qu'elle représente une chaîne de caractère (une suite de caractère), contrairement aux variables de type « int » qui représente un nombre entier.

Toujours avant le setup déclarer la variable BT_R de type char :

```
char BT_R;
```

Il est aussi nécessaire de définir le débit de transmission du signal Bluetooth comme nous l'avions fait pour le moniteur série dans le « Tutoriel_Arduino_Introduction ».

Dans le setup, rentrer la ligne :

```
BT.begin(9600);
```

Dans la boucle principale du programme nous allons alors chercher à lire l'information dès que celle-ci est reçu par le module.

Dans la loop, rentrer les lignes :

```
if (BT.available()) {
        BT_R = BT.read();
}
```

La variable BT_R prendra alors la valeur reçue par le module Bluetooth dès qu'une donnée sera envoyé à celui-ci.

Vous devez obtenir un programme comme ci-dessous.

```
#include <SoftwareSerial.h>

SoftwareSerial BT(10, 11);

char BT_R;

void setup() {

  BT.begin(9600);

}

void loop() {

  if (BT.available()) {

    BT_R = BT.read();

  }
}
```

Nous souhaitons maintenant lire sur le moniteur série les valeurs reçues par le module Bluetooth.

Compléter le programme en intégrant le moniteur série comme effectuer dans le « Tutoriel_Arduino_Introduction ». Il faut écrire :

- Une ligne d'initialisation du moniteur série dans le setup.
- Une ligne d'affichage de la valeur de BT_R dans le moniteur série après la lecture du module Bluetooth.

Il est temps de tester votre programme.

Téléverser le programme dans la carte Arduino.

Ouvrir l'application Android sur le smartphone.

Avec le smartphone, se connecter au Bluetooth de l'Arduino.

Appuyer sur les boutons « Allumer » et « Eteindre » et regarder le résultat sur le moniteur série de Arduino.

Si des caractères bizarres s'affiche dans le moniteur série, c'est normal puisqu'il lit directement l'octet envoyé. Pour convertir l'octet en nombre décimal il faut utiliser la fonction :

```
Serial.println(BT_R, DEC);
```

Le « DEC » permettra la conversion en décimal.

Il est maintenant possible de traiter l'information reçue pour contrôler la LED.

Compléter votre programme en ajoutant les lignes manquantes dans votre programme.

Les lignes 11 et 20 doivent déjà être complétées par le moniteur série.

Compléter les lignes 6, 12, 23 et 27 pour ajouter la LED de la borne 13 (LED de l'Arduino) au programme et allumer et éteindre la LED en fonction de la variable BT_R.

Tester votre programme et vérifier le fonctionnement.

Question 1 – Copier-coller votre programme complété sur le document Word.

```
1  #include <SoftwareSerial.h>
2
3  SoftwareSerial BT(10, 11);
4
5  char BT_R;
6  ..... // Déclaration de la variable LED égale à 13.
7
8  void setup() {
9
10   BT.begin(9600);
11   ..... // Initialisation du moniteur série.
12   ..... // La variable LED est paramétré comme une sortie.
13 }
14
15 void loop() {
16
17   if (BT.available()) {
18
19     BT_R = BT.read();
20     ..... // Affichage de la valeur de BT_R dans le moniteur série.
21
22     if (BT_R == 1) {
23       ..... // La LED s'allume.
24     }
25
26     if (BT_R == 0) {
27       ..... // La LED s'éteint.
28     }
29   }
30 }
```

DEUXIEME PROGRAMME

Ce deuxième programme permettra d'envoyer plusieurs informations en même temps à la carte Arduino via l'application Android et la transmission Bluetooth, de décoder ces données sur la carte Arduino puis de les renvoyer à l'application pour les afficher.

Les informations envoyées depuis l'application seront les valeurs de 3 curseurs que l'utilisateur peut faire varier grâce à l'application. Ces valeurs devront être envoyées en même temps sous la forme d'un texte.

Ces valeurs seront alors collectées par l'Arduino pour être traiter de façon à isoler chacune des 3 valeurs correspondants aux 3 curseurs.

La chaines de caractère à envoyer sera la suivante :

$$RGB,X_1,X_2,X_3,\backslash n$$

Avec X_1, X_2 et X_3 la valeur des 3 curseurs.

« \n » signifie que la chaîne de caractère s'arrête à cet endroit.

Du côté de l'Arduino nous enverrons les 3 valeurs reçues sous la forme :

$$X_1\ X_2\ X_3$$

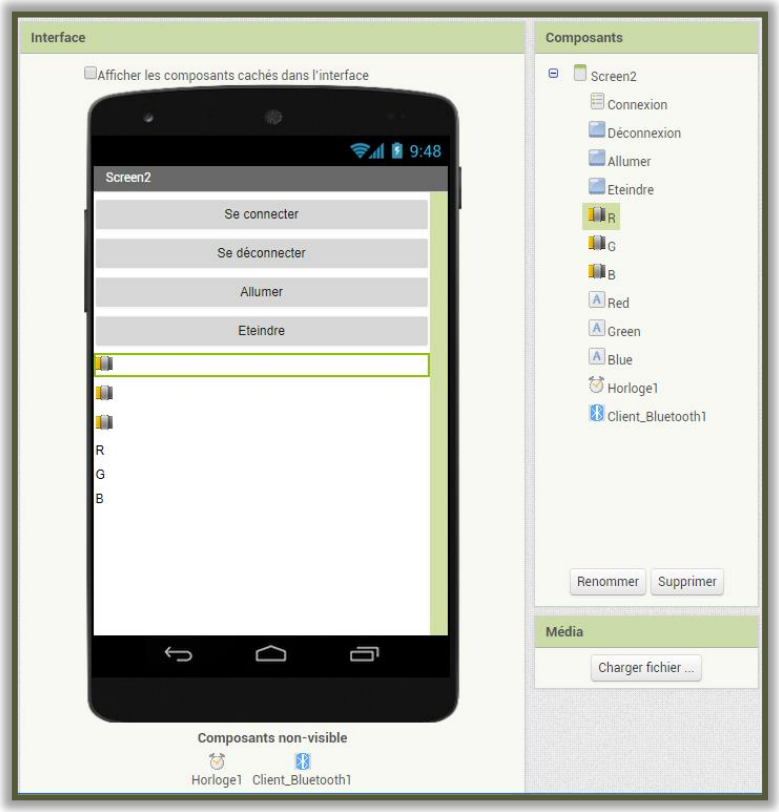
Avec un espace entre chaque valeur.

PROGRAMMATION APP INVENTOR

Pour ce deuxième exercice nous partirons de la base effectuée au premier exercice.

« Enregistrer sous » le projet « Premier_exercice » et le nommer « Deuxième_exercice ».

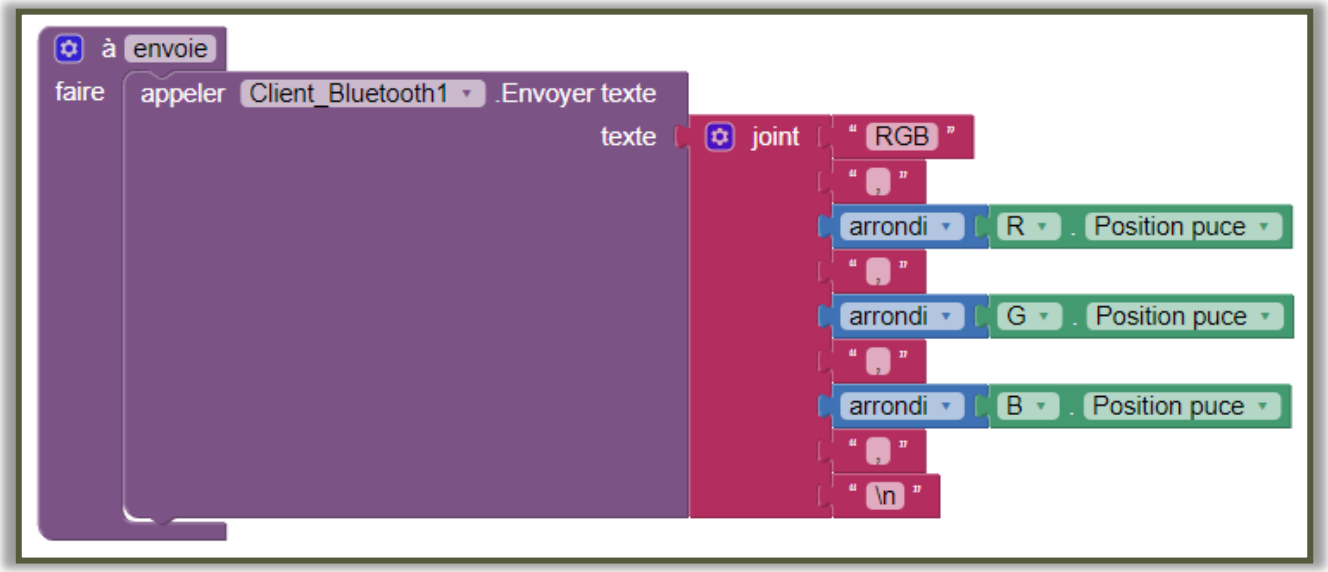
Compléter ensuite l'écran 2 pour obtenir la même interface que ci-dessous.



Deux fonctions vont être créée :

- Envoie : permet de mettre sur une chaine de caractère les données des 3 curseurs et de les envoyer.
- Réception : permet de recevoir les données de l'Arduino et de les afficher.

Créer la fonction « envoie »



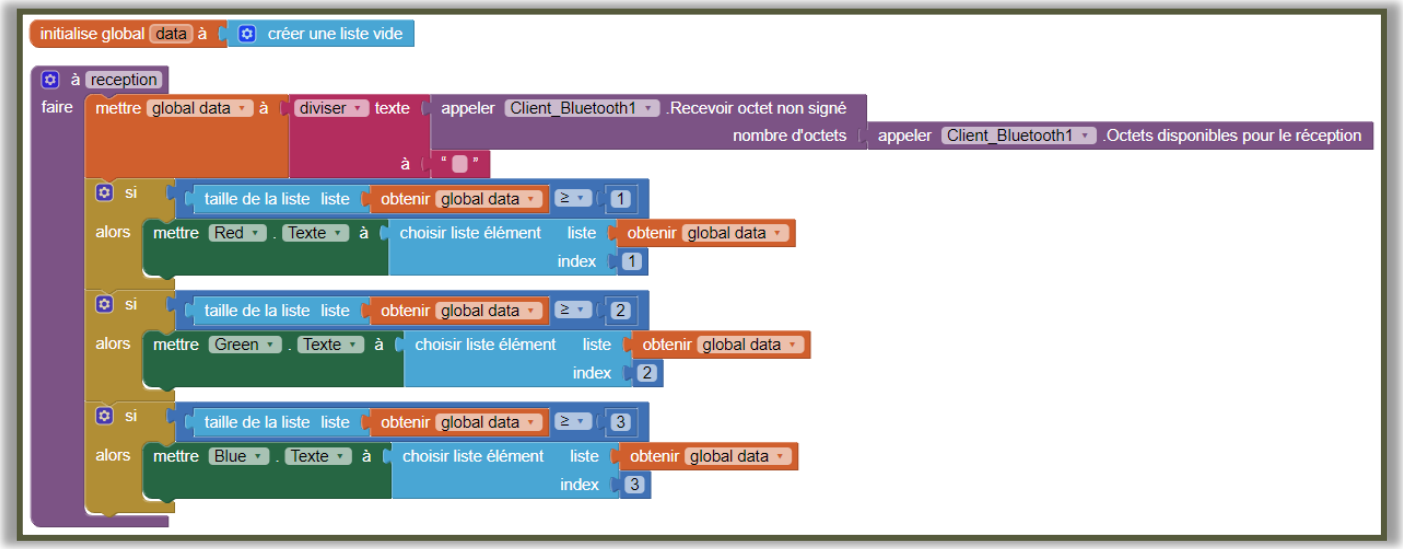
La fonction « envoie » permet d'envoyer un texte via le Bluetooth du smartphone. Le bloc « joint » permet de rattacher ensemble plusieurs textes pour arriver à la forme voulue :

$$\mathbf{RGB,X_1,X_2,X_3,\backslash n}$$

Avec X_1, X_2 et X_3 la valeur des 3 curseurs.

« \n » signifie que la chaîne de caractère s'arrête à cet endroit.

Créer la fonction « reception »

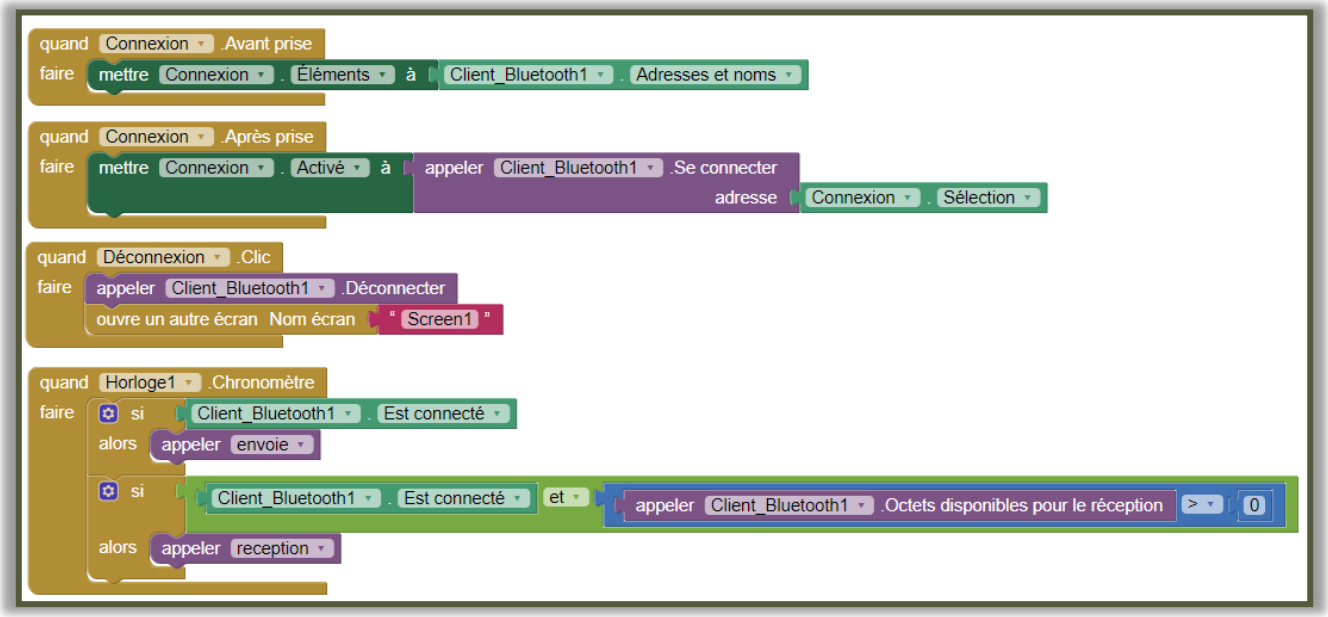


Pour traiter la chaîne de caractère envoyé par l'Arduino il suffit de diviser cette chaîne en plusieurs éléments composant une liste et correspondant au valeur X_1, X_2 et X_3. Il faut alors définir les espaces comme étant les

endroits où diviser la chaine de caractère. De ce fait, la valeur X_1 sera à l'index 1 de la liste, X_2 à l'index 2 et X_3 à l'index 3.

L'horloge permettra d'envoyer les données souhaitées à une certaine cadence. Ici on enverra les données toutes les 100 millisecondes.

Créer les blocs suivants.



PROGRAMMATION ARDUINO

Le programme Arduino pour réceptionner et envoyer les données via le Bluetooth est assez complexe. Vous trouverez donc le programme déjà rédigé sur le site du professeur.

Téléverser le programme Arduino dans la carte et vérifier son bon fonctionnement avec l'application Android et le moniteur série.

Observer le résultat et analyser le programme Arduino.

Sur l'application on observe qu'il y a une parenthèse « (» avant la valeur de R et une parenthèse «) » après la valeur de B.

Question 2 - Trouver un moyen de supprimer ses parenthèses. Décrire votre solution.