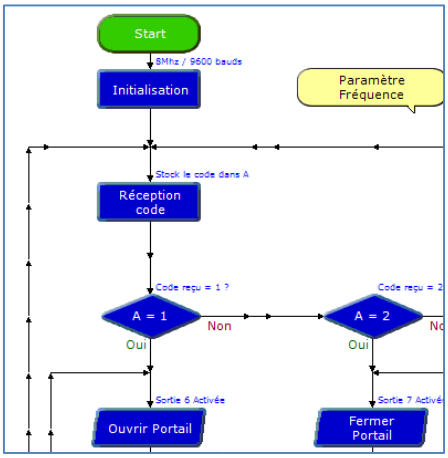
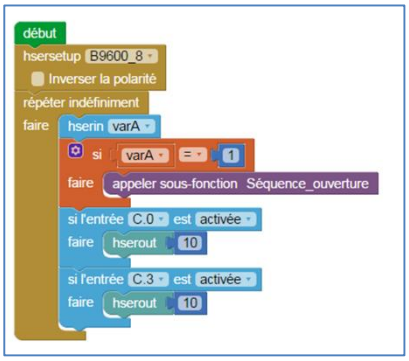
Option module Bluetooth pour Portail Coulissant

Application Android avec App Inventor
Programmes PICAXE, Blockly et Arduino

3 modes de programmation possibles



Logigrammes



Blocs

```
...
//Programme principal
//Boucle infinie
void loop()
{
 if (Serial.available() > 0) // Y a-t-il de l'activité sur le port série ?
 {
  commande_recue = Serial.read();
  if (commande_recue == 1) // Rentrer dans la boucle si la c
  {
   if (digitalRead(fdc_ouverture)==LOW)
   {
    do
    {
     digitalWrite(ouverture, HIGH); // activation du moteur : sens ou
     digitalWrite(gyrophare, HIGH); // déclenchement du gyrophare
    }while (digitalRead(fdc_ouverture)==LOW); // rester dans la bo

   digitalWrite(ouverture, LOW);
   digitalWrite(fermeture, LOW); // arrêt du moteur
   digitalWrite(gyrophare, LOW); // extinction du gyrophare
  }
```

Textuelle

A4 TECHNOLOGIE
www.a4.fr

PICAXE est une marque de la Sté Revolution Education.
AutoProg est un système développé par la Sté A4, qui utilise des microcontrôleurs PICAXE.

Environnement de programmation PICAXE qui intègre la programmation graphique de type logigrammes (Logicator) et la programmation par blocs (Blockly).

Le design matériel de l'Arduino est distribué sous licence Creative Commons et est disponible sur le site d'Arduino.
Le code source de l'environnement de programmation et les bibliothèques embarquées sont disponibles sous licence GNU.
AutoProgUno est un système développé par la Sté A4, qui utilise la carte Arduino UNO.

APP INVENTOR est un environnement de programmation orientée objet, accessible aux non-initiés pour concevoir des applications Android.

L'ensemble des ressources numériques disponibles autour de nos projets et maquettes sont téléchargeables librement et gratuitement sur **www.a4.fr**

Décembre 2015
André Bernot, Christophe Mahier, Julien Launay

Edité par la société A4 Technologie
Tél. : 01 64 86 41 00 - Fax : 01 64 46 31 19
www.a4.fr

SOMMAIRE

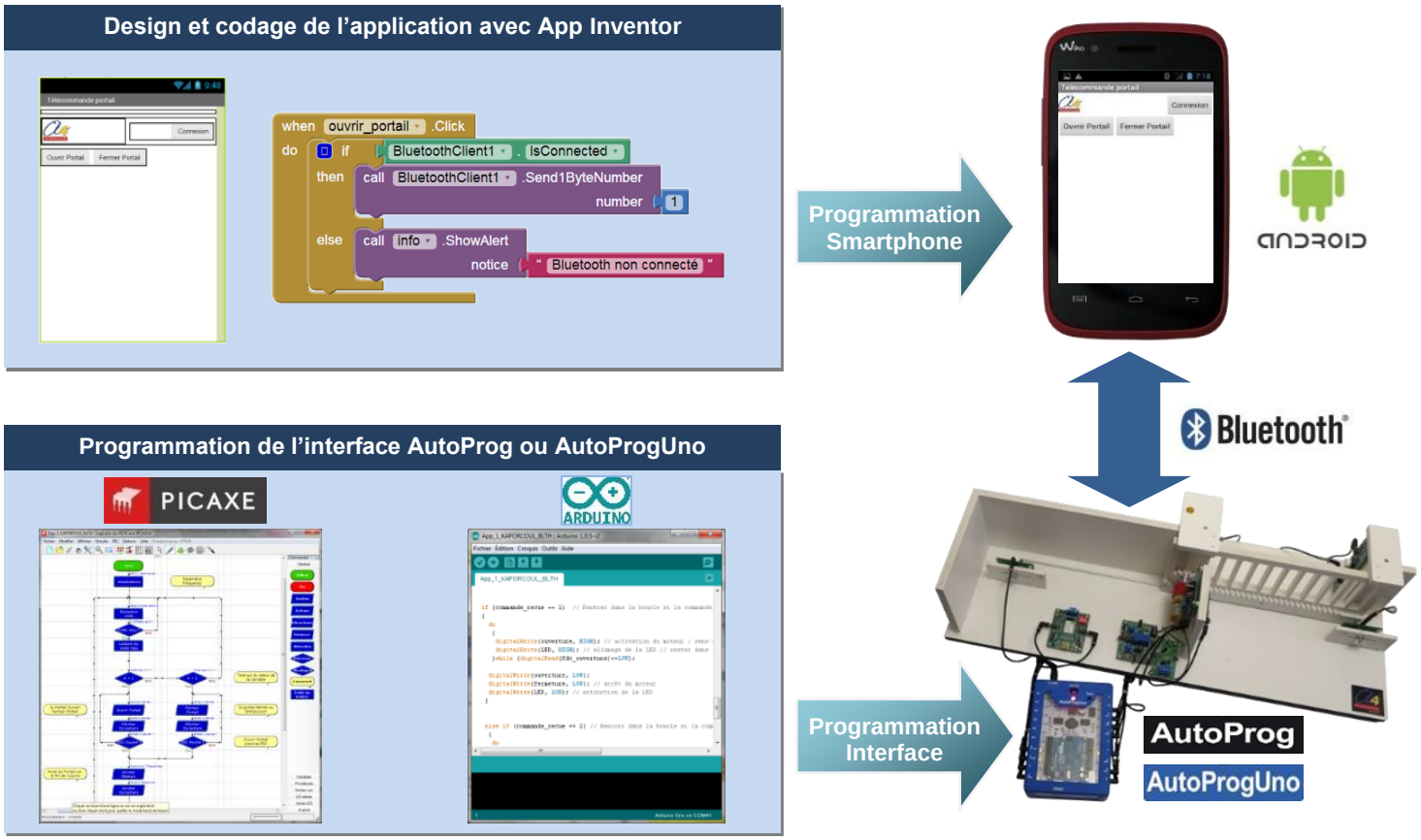
1. Introduction

Ce document illustre l'utilisation du module Bluetooth au travers d'exemples basés sur la maquette du **Portail Coulissant** d'A4 équipée avec l'**option module Bluetooth**.
Les applications proposées permettent d'interagir entre une application chargée sur **Smartphone Android** et la maquette pilotée par une interface programmable **AutoProg** ou **AutoProgUno**.

Les programmes sont développés avec les IDE* :

- **App Inventor 2** pour les applications Android ;
- **PICAXE Logicator** ou **PICAXE Editor 6** pour les applications AutoProg ;
- **Arduino** pour les applications AutoProgUno.

** IDE : Environnement de Développement Intégré*



1.1. Prérequis souhaitables

La mise en œuvre des applications suppose que l'utilisateur ait des notions de base autour des logiciels et matériels utilisés. Il est utile de maîtriser la programmation de base des interfaces AutoProg ou AutoProgUno pour piloter le portail coulissant.

1.2. Organisation de ce dossier

Les applications se présentent sous forme de fiches classées par ordre de difficulté croissante.
Elles permettent de découvrir la manière d'établir une communication bidirectionnelle entre le smartphone et l'interface de pilotage du portail.
Des propositions de modifications sont suggérées à la fin de chaque fiche.
On notera que les applications peuvent être transposées aisément vers d'autres maquettes d'automatisme de notre gamme.
Les éléments ou briques de programmes qui permettent d'assurer la communication en Bluetooth entre le smartphone et l'interface sont similaires d'une application à l'autre.
Ces briques peuvent apparaître complexes, cependant leur maîtrise n'est pas indispensable pour appréhender les applications.
L'annexe de ce dossier propose des explications complémentaires autour des applications.

2. Éléments nécessaires

2.1. Matériels

- Maquette de portail coulissant BE-APORCOUL.
- Module Bluetooth K-AP-MBLTH.
- Interface programmable AutoProg ou AutoProgUno avec son câble de programmation.
- Onze câbles de liaison jack compatibles AutoProg / AutoProgUno pour établir les liaisons entre l'interface programmable et la maquette.
- Smartphone ou tablette Android équipé du Bluetooth + câble USB de liaison au PC.

2.2. Logiciels

L'ensemble des logiciels utilisés est GRATUIT.

- IDE App Inventor 2 (http://appinventor.mit.edu/explore/) pour la programmation des applications pour smartphone.
  L'environnement fonctionne sur le cloud. Il est hébergé sur un serveur.
  Un compte Gmail est nécessaire pour l'utiliser.
  Cliquer sur le bouton **Create** en haut à droite de la page d'accueil d'App Inventor 2 pour lancer l'IDE.
- IDE PICAXE Logicator ou PICAXE Editor6 (http://www.picaxe.com/Software) pour la programmation de l'interface AutoProg.
- IDE Arduino (http://arduino.cc/en/Main/Software) pour la programmation de l'interface AutoProgUno

2.3. Ressources complémentaires

Des ressources complémentaires sont disponibles sur www.a4.fr :

- le dossier technique et pédagogique du Portail Coulissant pour la mise en œuvre de la maquette ;
- le dossier App Inventor 2 pour prendre en main AppInventor 2 ;
- le guide d'utilisation PICAXE Logicator ;
- le dossier du module Bluetooth.

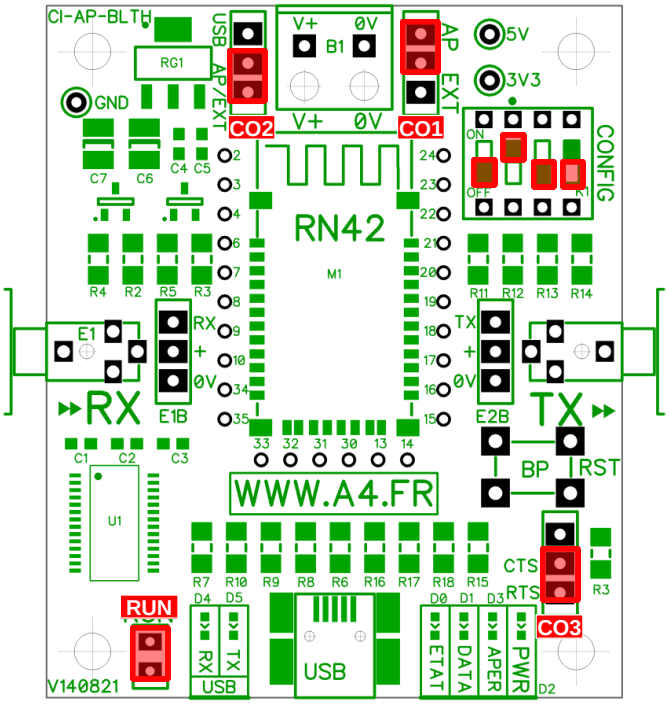
D'autres ressources sont disponibles sur internet. Vous pouvez entrer les mots-clés suivants pour les localiser avec un moteur de recherche : « Tuto app inventor 2 », « Tuto picaxe », « Tuto arduino uno ».

3. Le module Bluetooth

Le module Bluetooth développé par A4 Technologie permet de convertir le protocole Bluetooth en protocole de communication type Série qui est le mode de communication classique utilisé avec PICAXE ou Arduino.
Ce module accepte différentes configurations.
En mode avancé, il peut être configuré au travers d'une liaison par connexion USB à un PC ou par l'envoi de commandes au travers de ses liaisons RX et TX.
La documentation technique du module Bluetooth décrit en détail les fonctionnalités du module.
Elle est téléchargeable sur www.a4.fr

3.1. Configuration

Positionner les cavaliers et interrupteurs comme indiqué par les positions repérées en rouge ci-dessous.



- Le cavalier repéré **RUN** est utilisé lors de la mise au point de programmes avec **Arduino**. Il doit être ôté pour permettre le téléversement du programme puis doit être remis lors de l'utilisation.
- La mise au point de programmes avec **PICAXE** ne nécessite pas d'ôter ce cavalier pour transférer le programme.
- Les cavaliers **CO1** et **CO2** permettent de sélectionner le mode d'alimentation du module Bluetooth. Dans la configuration ci-dessus, son alimentation provient directement de l'interface AutoProg ou AutoProgUno au travers des cordons de liaison avec le module ; ils sont positionnés respectivement sur AP et sur AP/EXT.
- Le cavalier **CO3** est utilisé en mode avancé pour relier ou dissocier les signaux CTS et RTS nécessaires au fonctionnement du module Bluetooth. Ici, il est positionné sur CTS/RTS.
- Les interrupteurs **CONFIG** permettent de paramétrer le mode de fonctionnement du module Bluetooth. Ici, l'interrupteur n°2 est positionné sur ON pour sélectionner une vitesse de transmission des données à 9600 bauds.

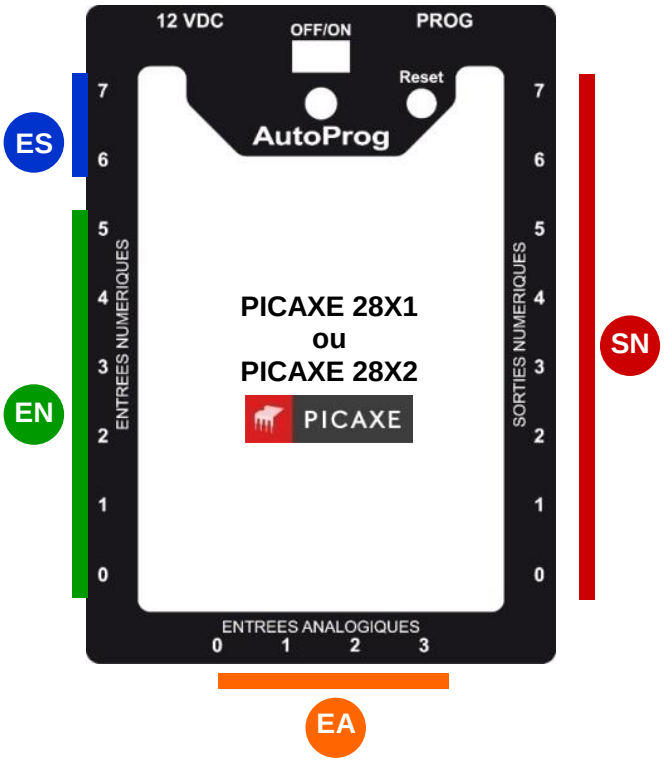
3.2. Témoins lumineux

PWR indique que le module est sous tension.
APER indique que le module est associé avec un matériel Bluetooth.
DATA indique qu'il y a un flux de données entre le module et l'appareil avec lequel il est connecté.
ETAT indique que le module est opérationnel. L'affichage clignotant indique qu'il n'est pas opérationnel.
USB RX indique qu'il y a un flux de données sur la liaison USB du PC vers le module.
USB TX indique qu'il y a un flux de données sur la liaison USB du module vers le PC.

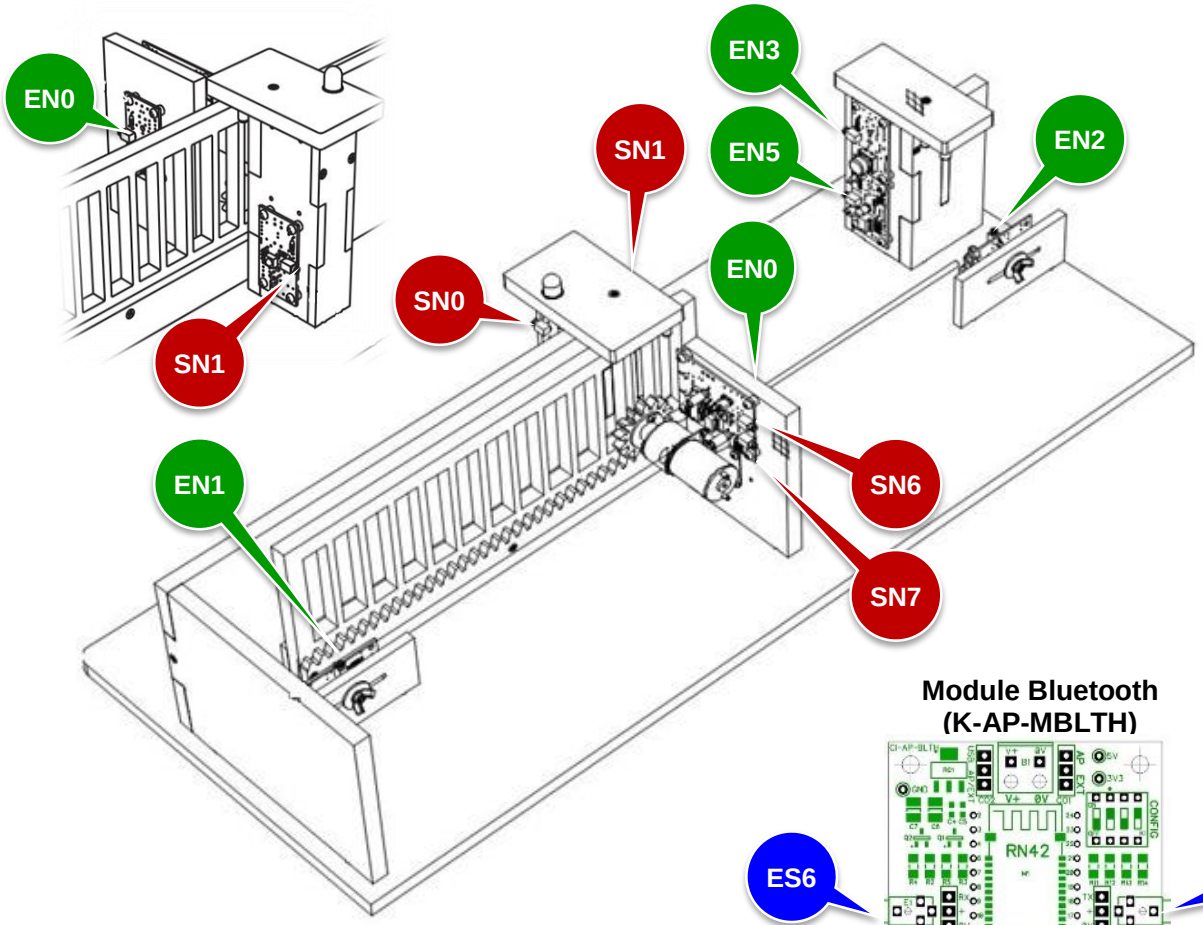
4. Connexion de l'interface AutoProg (PICAXE) à la maquette

4.1. Affectation des entrées / sorties de l'interface AutoProg (PICAXE)

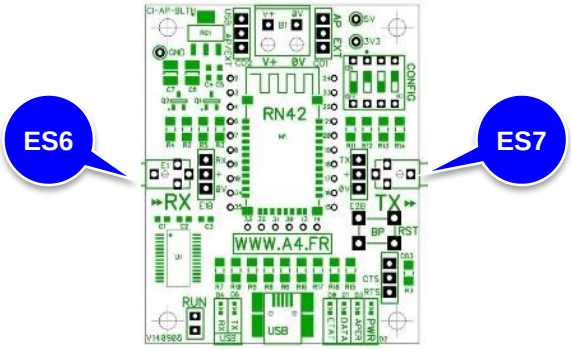
ES	**MODULE DE COMMUNICATION POUR ENTRÉES / SORTIES NUMÉRIQUES**
7	Bluetooth connecteur TX (liaison avec l'entrée de données HSERIN)
6	Bluetooth connecteur RX (liaison avec la sortie de données HSEROUT)
EN	**MODULES CAPTEURS POUR ENTRÉES NUMÉRIQUES**
5	Récepteur infrarouge
4	*(libre)*
3	Bouton-poussoir extérieur
2	Fin de course portail fermé
1	Fin de course portail ouvert
0	Bouton-poussoir intérieur
EA	**MODULES CAPTEURS POUR ENTRÉES ANALOGIQUES**
3	*(libre)*
2	*(libre)*
1	*(libre)*
0	*(libre)*
SN	**MODULES ACTIONNEURS SORTIES NUMÉRIQUES**
7	Moteur sens Fermeture
6	Moteur sens Ouverture
5	*(libre)*
4	*(libre)*
3	*(libre)*
2	*(libre)*
1	Emetteur infrarouge
0	Gyrophare



4.2. Schéma de câblage de la maquette avec l'interface AutoProg



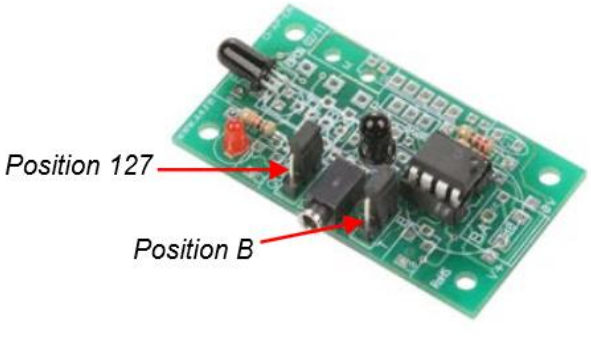
Module Bluetooth (K-AP-MBLTH)



RAPPEL

Configuration du module moteur :
Le cavalier de configuration est sur la position **INT** par défaut (alimentation du moteur par AutoProg).

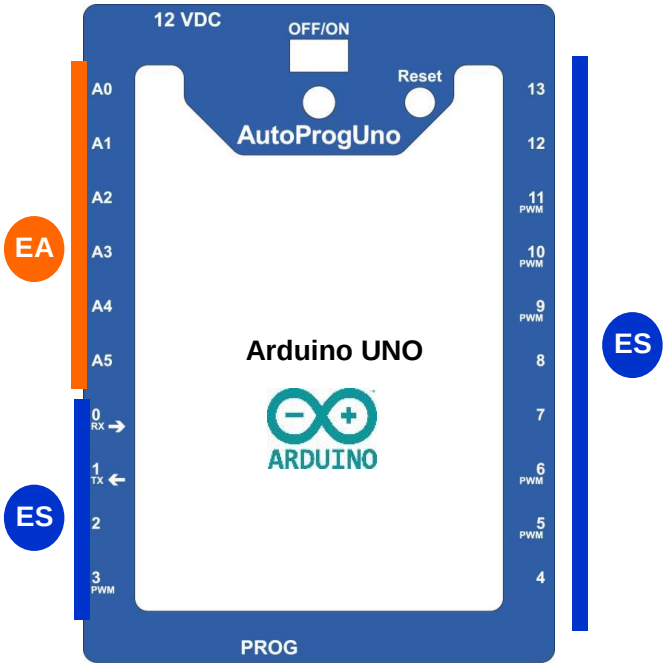
Configuration du module récepteur infrarouge :
Les cavaliers de configuration sont sur les positions **B et 127** (mode barrière IR).



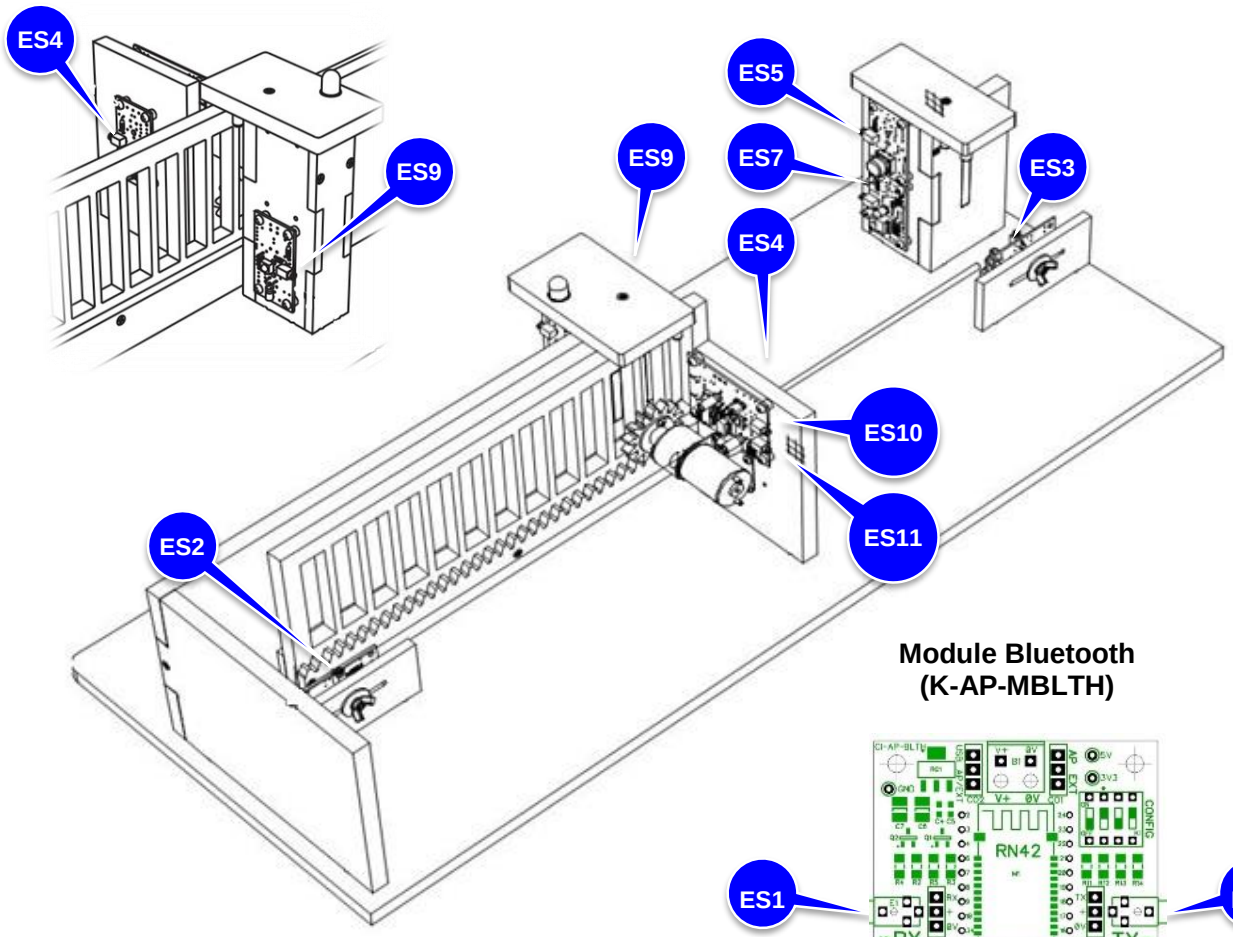
5. Connexion de l'interface AutoProgUno (Arduino) à la maquette

5.1. Affectation des entrées / sorties de l'interface AutoProgUno (Arduino)

EA	**MODULES CAPTEURS POUR ENTRÉES ANALOGIQUES**
A5	*(libre)*
A4	*(libre)*
A3	*(libre)*
A2	*(libre)*
A1	*(libre)*
A0	*(libre)*
ES	**MODULES CAPTEURS OU ACTIONNEURS POUR ENTRÉES / SORTIES NUMÉRIQUES**
13	*(libre)*
12	Gyrophare
11	Moteur sens Fermeture
10	Moteur sens Ouverture
9	Émetteur infrarouge
8	*(libre)*
7	Récepteur infrarouge
6	*(libre)*
5	Bouton-poussoir extérieur
4	Bouton-poussoir intérieur
3	Fin de course portail fermé
2	Fin de course portail ouvert
1	Bluetooth connecteur RX (TX AutoProgUno)
0	Bluetooth connecteur TX (RX AutoProgUno)



5.2. Schéma de câblage de la maquette avec l'interface AutoProgUno

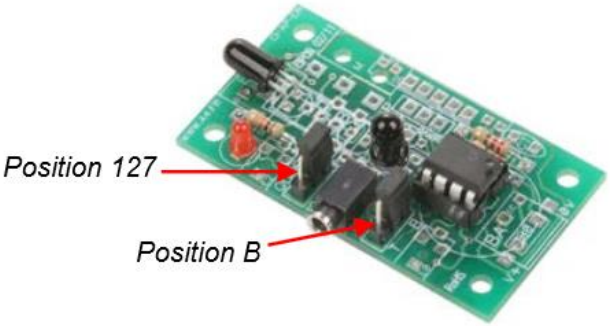


Module Bluetooth (K-AP-MBLTH)

RAPPEL

Configuration du module moteur :
Le cavalier de configuration est sur la position **INT** par défaut (alimentation du moteur par d'AutoProg)

Configuration du module récepteur infrarouge :
Les cavaliers de configuration sont sur les positions **B et 127** (mode barrière IR).



6. Mise en service des applications

Les applications proposées dans les fiches suivantes sont créées avec différents IDE (Environnement de Développement Intégré).
Pour chaque application, un ensemble de fichiers à charger dans le smartphone Android et dans l'interface programmable AutoProg ou AutoProgUno est proposé.
Une description et des explications sont proposées pour chaque application.
Les éléments clés sont mis en évidence et des suggestions de modification sont proposées en vue d'adapter les programmes à un nouveau contexte d'utilisation.
Il est indispensable que la fonction Bluetooth soit mise en service dans les paramètres du smartphone ou de la tablette Android.

6.1. Programmes pour smartphone Android

MIT App Inventor	http://appinventor.mit.edu/explore/ Cet IDE est gratuit. Il fonctionne sur le cloud. Il ne nécessite pas d'installation en local. Un compte Gmail est indispensable pour accéder à son espace de développement.
Matériel associé	- Smartphone ou tablette Android - Cordon de liaison USB avec le PC pour transférer les programmes dans le smartphone.
Fichiers	Fichiers .**APK** (application) à copier et à installer dans le smartphone. Fichiers .**AIA** (code source) à ouvrir dans App Inventor 2.
Notes	Un guide de prise en main de l'IDE App Inventor 2 est proposé en téléchargement sur www.a4.fr La page d'accueil d'App Inventor propose un nombre important de tutoriels (en anglais). Le mot-clé « Tuto App Inventor » tapé dans un moteur de recherche renvoie vers de nombreuses ressources autour d'App Inventor.

6.2. Programmes pour AutoProg

PICAXE	**PICAXE Logicator** http://www.picaxe.com/Software/PICAXE/Logicator-for-PICAXE/ ou **PICAXE Editor6** http://www.picaxe.com/Software/PICAXE/PICAXE-Editor-6/ Cet IDE est gratuit. Il doit être installé sur un PC.
Matériel associé	- Interface programmable AutoProg ou autre carte PICAXE compatible. - Cordon de programmation PICAXE USB pour transférer les programmes dans l'interface AutoProg.
Fichiers	Fichiers .**PLF** à ouvrir dans l'IDE PICAXE et à transférer dans l'interface.
Notes	www.a4.fr propose de nombreuses ressources autour de PICAXE ; notamment le Guide d'utilisation de Logicator et les dossiers AutoProg et AutoProgUno. Il existe d'autres IDE compatibles PICAXE (par exemple Scratch).

6.3. Programmes pour AutoProgUno

ARDUINO	http://arduino.cc/en/Main/Software (rubrique Arduino IDE) Cet IDE est gratuit. Il doit être installé sur un PC.
Matériel associé	- Interface programmable AutoProgUno ou autre carte Arduino compatible. - Cordon de liaison USB type imprimante avec le PC pour transférer les programmes dans l'interface AutoProgUno.
Fichiers	Fichiers .**INO** à ouvrir dans l'IDE Arduino et à téléverser dans l'interface.
Notes	Les programmes proposés dans ce document sont développés en C avec l'IDE standard Arduino. Il existe d'autres IDE compatibles Arduino (par exemple Scratch ou Ardublock).

7. Fiche technique N°1 - Télécommande 2 boutons

But de l'application

Piloter à distance l'ouverture et la fermeture du portail coulissant avec un smartphone.

Notions de programmation abordées

Envoi d'une commande du smartphone vers l'interface de pilotage du portail.
Réception et traitement d'une commande reçue par le module Bluetooth.

Fichiers téléchargeables sur www.a4.fr

Application Android	***F1_PCOUL.apk***
Code source App Inventor2	***F1_PCOUL.aia***
Code source Arduino Uno	***F1_PCOUL.ino***
Code source PICAXE Logicator	***F1_PCOUL_LG.plf***
Code source PICAXE Editor 6	***F1_PCOUL_PE6.plf***
Code source PICAXE Logicator pour Autoprog 28X1	***F1_PCOUL_LG_28X1.plf***
Code source PICAXE Editor 6 pour Autoprog 28X1	***F1_PCOUL_PE6_28X1.plf***
Code source PICAXE Blockly	***F1_PCOUL.xml***

7.1. Fonctionnement de l'application

Interface utilisateur sur le smartphone

Le bouton **Connexion** permet d'établir la connexion entre le smartphone et le module Bluetooth.

Le bouton **Ouvrir Portail** déclenche l'ouverture du portail par envoi du code « 1 ».

Le bouton **Fermer Portail** déclenche la fermeture du portail par envoi du code « 2 ».

Programme d'automatisme

Le programme de pilotage de la maquette déclenche l'ouverture du portail lorsque le code « 1 » est reçu et le ferme si le code « 2 » est reçu.

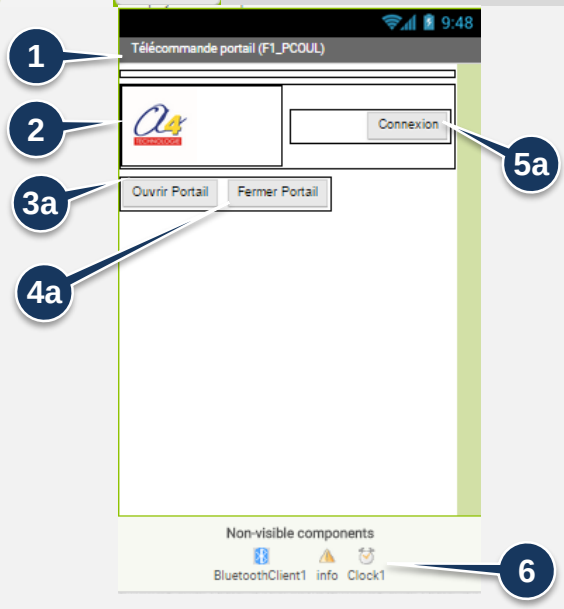
Suggestions de modifications

Interface utilisateur
Dans la partie **Designer** d'App Inventor, modifier les **propriétés** des composants de l'interface utilisateur ; changer le titre de l'application, remplacer les boutons standards par une image personnalisée, etc.

Programmes
Modifier le code envoyé par le smartphone et adapter le programme d'automatisme en conséquence.

Programme App Inventor **F1_PCOUL.aia**

IDE App Inventor2 (fenêtre Designer)



1 Titre de l'application.

2 Image insérée dans l'application.

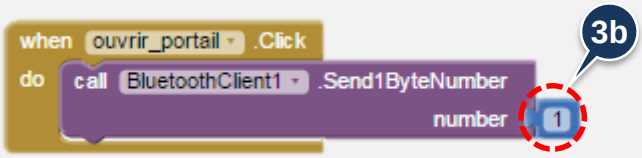
3a Déclenchement de l'ouverture en envoyant le code **« 1 ».**

4a Déclenchement de la fermeture en envoyant le code **« 2 ».**

5a Gestion de la communication en Bluetooth.

6 Éléments non visibles de l'application.
Des explications sont données en Annexe.

IDE App Inventor2 (fenêtre Blocks)



3b **Déclenchement de l'ouverture**

Lorsque vous cliquez sur le bouton nommé **ouvrir_portail** :

Le module Bluetooth envoye le code « **1** » par liaison Bluetooth.

4b **Déclenchement de la fermeture**

Lorsque vous cliquez sur le bouton nommé **fermer_portail** :

Le module Bluetooth envoye le code « **2** » par liaison Bluetooth.

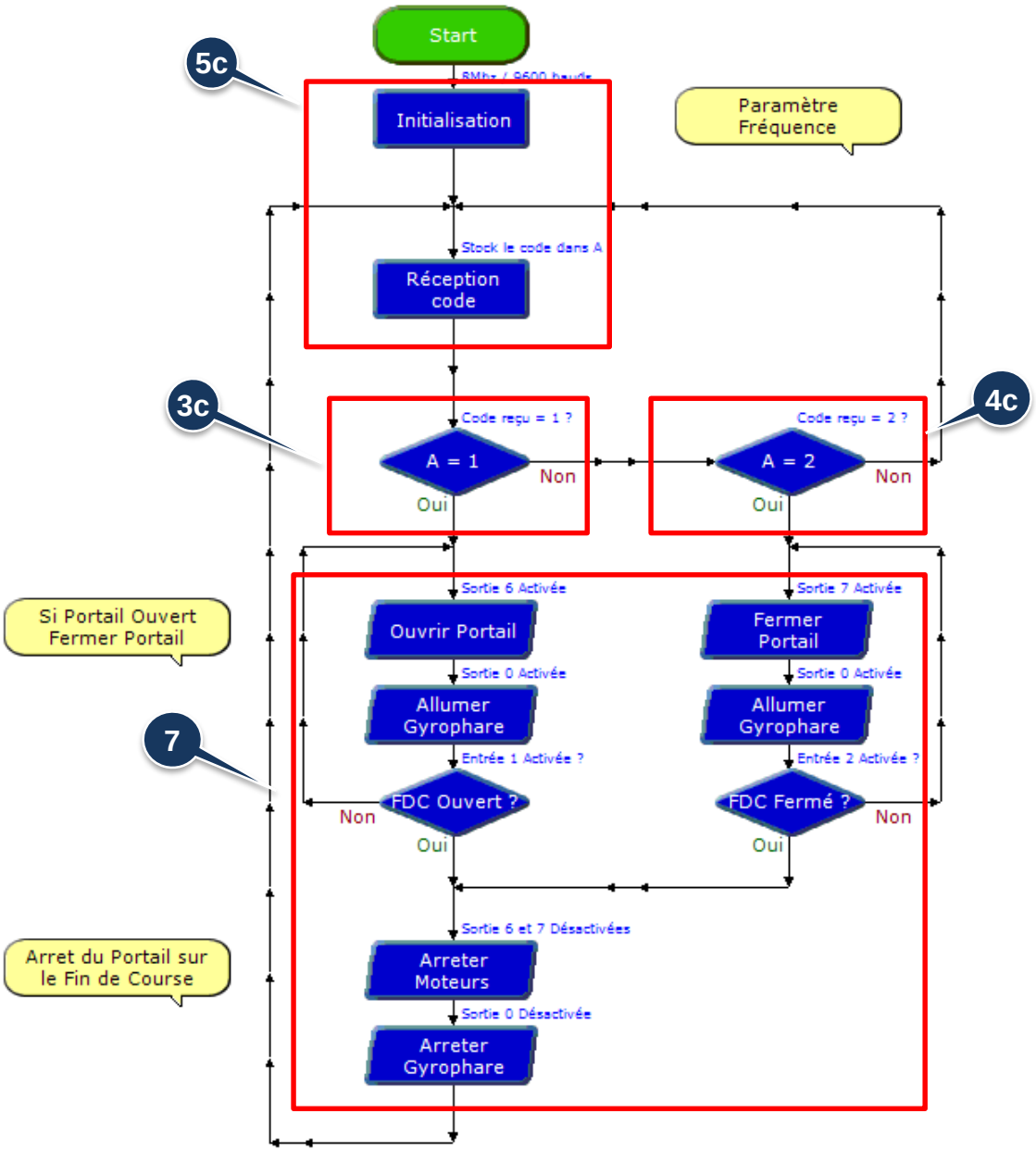
5b **Gestion de la communication en Bluetooth**

Ces blocs sont nécessaires pour gérer la communication en Bluetooth.
Ils sont communs à toutes les applications proposées dans ce dossier.
Des explications supplémentaires sont proposées en Annexe.

Note : on peut déployer la visualisation de ces blocs dans App Inventor 2 en effectuant un clic droit sur le bloc souhaité et en sélectionnant « Expand Block ».

Programme PICAXE **F1_PCOUL.plf**

PICAXE **IDE Logicator ou PICAXE Editor 6**

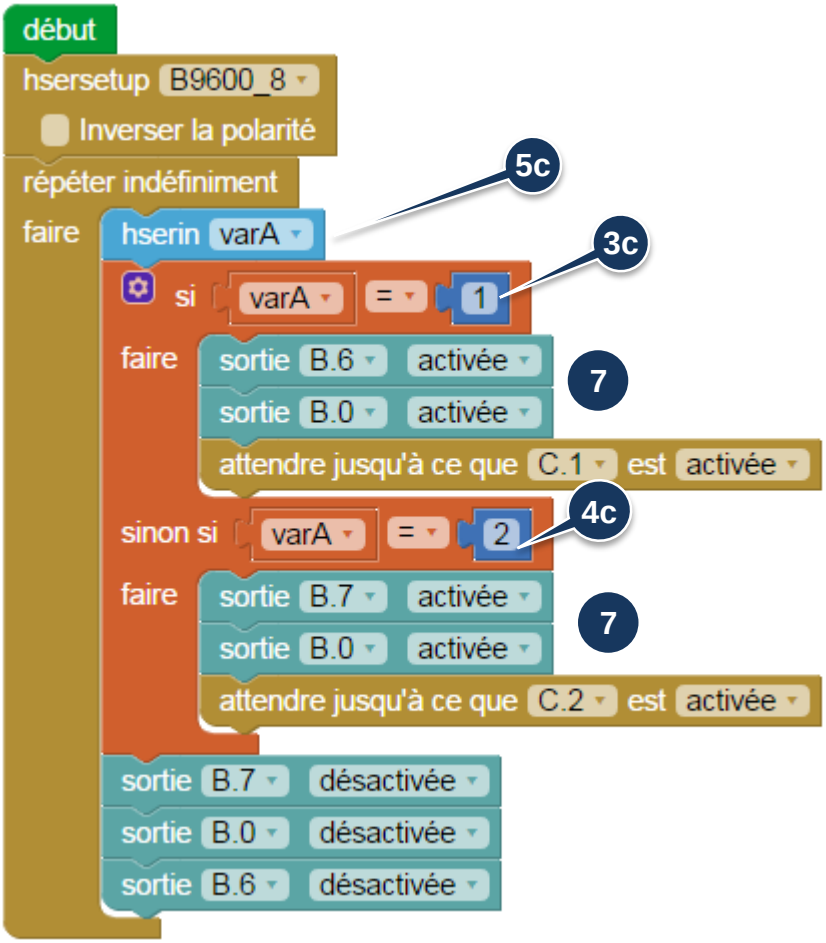


3c Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

4c Vérification du code reçu pour la demande de fermeture du portail (code « 2 »).

5c Gestion et attente de réception des données émises par le smartphone.

7 Gestion de l'ouverture et de la fermeture du portail.



3c Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

4c Vérification du code reçu pour la demande de fermeture du portail (code « 2 »).

5c Gestion et attente de réception des données émises par le smartphone.

7 Gestion de l'ouverture et de la fermeture du portail.

Programme Arduino F1_PCOUL.ino

IDE Arduino

(extrait du programme)

```
...
//Programme principal
//Boucle infinie
void loop()
{
 if (Serial.available() > 0) // Y a-t-il de l'activité sur le port série ?
 {
  commande_recue = Serial.read();

  if (commande_recue == 1)  // Rentrer dans la boucle si la commande_reçue est '1'
  {
    do                                    3d
     {
      digitalWrite(ouverture, HIGH); // activation du moteur : sens ouverture
      digitalWrite(gyrophare, HIGH); // déclenchement du  gyrophare
     }while (digitalRead(fdc_ouverture)==LOW); // rester dans la boucle tant que fdc_ ouverture n'est pas activé


    digitalWrite(ouverture, LOW);
    digitalWrite(fermeture, LOW); // arrêt du moteur
    digitalWrite(gyrophare, LOW); // extinction du gyrophare
   }

   else if (commande_recue == 2) // Rentrer dans la boucle si la commande_recu est '2'
    {                                    4d
     do
      {
        digitalWrite(fermeture, HIGH); // activation du moteur : sens ouverture
        digitalWrite(gyrophare, HIGH); // allumage du gyrophare
      } while (digitalRead(fdc_fermeture)==LOW); // rester dans la boucle tant que fdc_ ouverture n'est pas activé


    digitalWrite(ouverture, LOW);
    digitalWrite(fermeture, LOW); // arrêt du moteur
    digitalWrite(gyrophare, LOW); // extinction du gyrophare
    }

  }

 }
// Fin du Programme principal
...
```

3d Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

4d Vérification du code reçu pour la demande de fermeture du portail (code « 2 »).

8. Fiche technique N°2- Télécommande 1 boutons

But de l'application

Piloter l'ouverture et la fermeture du portail coulissant à distance avec un smartphone.

Notions de programmation abordées

Envoi d'une commande du smartphone vers l'interface de pilotage du portail.
Réception et traitement d'une commande reçue par le module Bluetooth.

Fichiers téléchargeables sur www.a4.fr

Application Android	***F2_PCOUL.apk***
Code source App Inventor2	***F2_PCOUL.aia***
Code source Arduino Uno	***F2_PCOUL.ino***
Code source PICAXE Logicator	***F2_PCOUL_LG.plf***
Code source PICAXE Editor 6	***F2_PCOUL_PE6.plf***
Code source PICAXE Logicator pour Autoprog 28X1	***F2_PCOUL_LG_28X1.plf***
Code source PICAXE Editor 6 pour Autoprog 28X1	***F2_PCOUL_PE6_28X1.plf***
Code source PICAXE Blockly	***F2_PCOUL.xml***

8.1. Fonctionnement de l'application

Interface utilisateur sur le smartphone

Le bouton **Connexion** permet d'établir la connexion entre le smartphone et le module Bluetooth.

Le bouton **Commande Portail** déclenche l'ouverture ou la fermeture du portail par envoi du code « 1 ».

Programme d'automatisme

En fonction de la position initiale du portail (ouvert ou fermé), le programme de pilotage de la maquette déclenche l'ouverture ou la fermeture du portail lorsque le code « 1 » est reçu.

Suggestions de modifications

Interface utilisateur
Ajouter un bouton **Stop** qui enverra le code « **2** ».

Programmes
Implémenter la fonction d'arrêt immédiat du portail lorsque l'utilisateur clique sur le bouton **Stop**.

Programme App Inventor **F2_PCOUL.aia**

IDE App Inventor2 (fenêtre Designer)

1 Titre de l'application.

2 Image insérée dans l'application.

3a Déclenchement de l'ouverture ou de la fermeture en envoyant le code **« 1 ».**

4a Gestion de la communication en Bluetooth.

5 Éléments non visibles de l'application.
Des explications sont données en Annexe.

IDE App Inventor2 (fenêtre Blocks)

3b

Déclenchement de l'ouverture ou de la fermeture

Lorsque vous cliquez sur le bouton nommé **ouvrir_portail** :

Le module Bluetooth envoye le code « **1** » par liaison Bluetooth.

4a

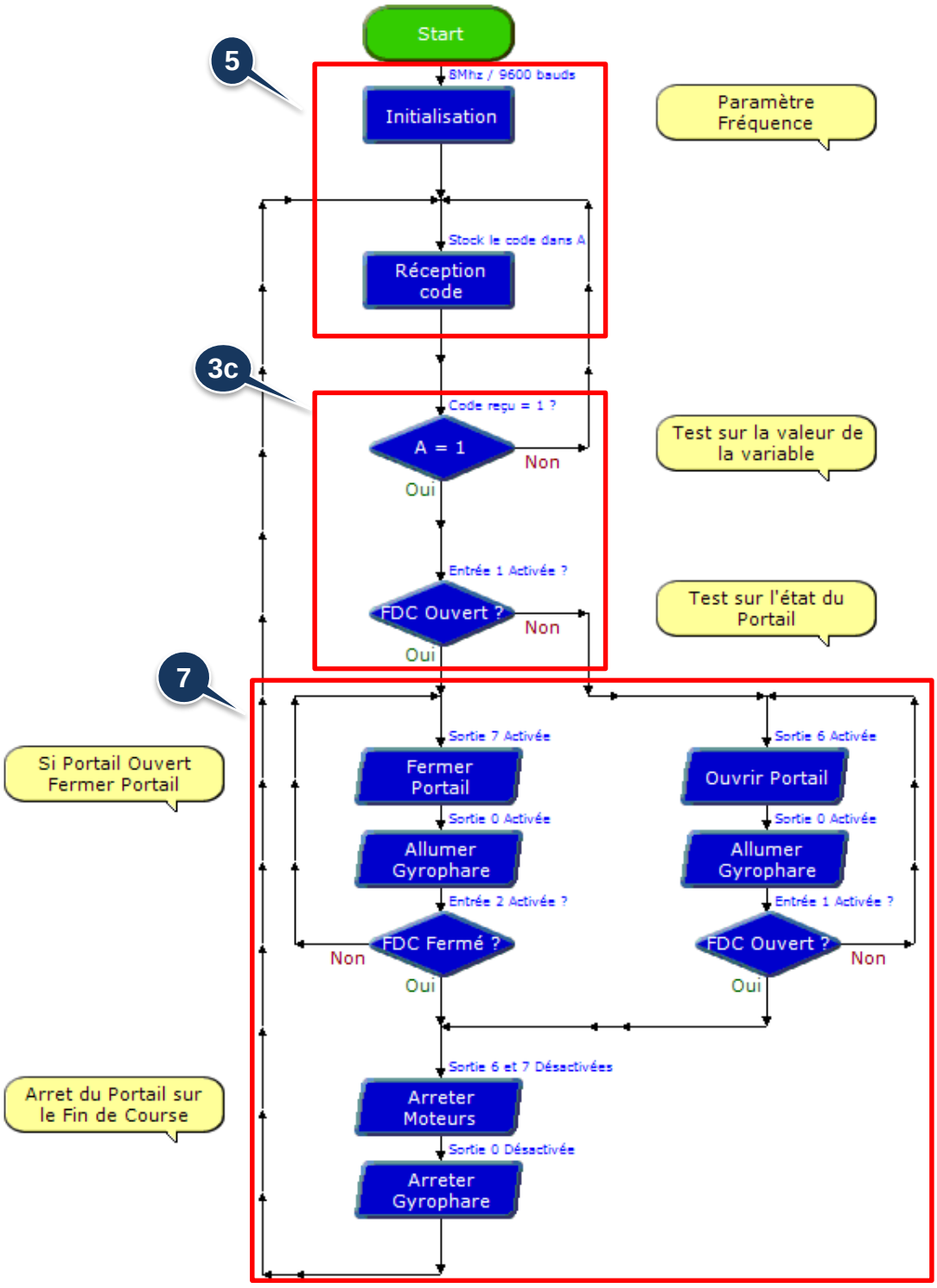
Gestion de la communication en Bluetooth

Ces blocs sont nécessaires pour gérer la communication en Bluetooth.
Ils sont communs à toutes les applications proposées dans ce dossier.
Des explications supplémentaires sont proposées en Annexe.

Note : on peut déployer la visualisation de ces blocs dans App Inventor 2 en effectuant un clic droit sur le bloc souhaité et en sélectionnant « Expand Block »

Programme PICAXE F2_PCOUL.plf

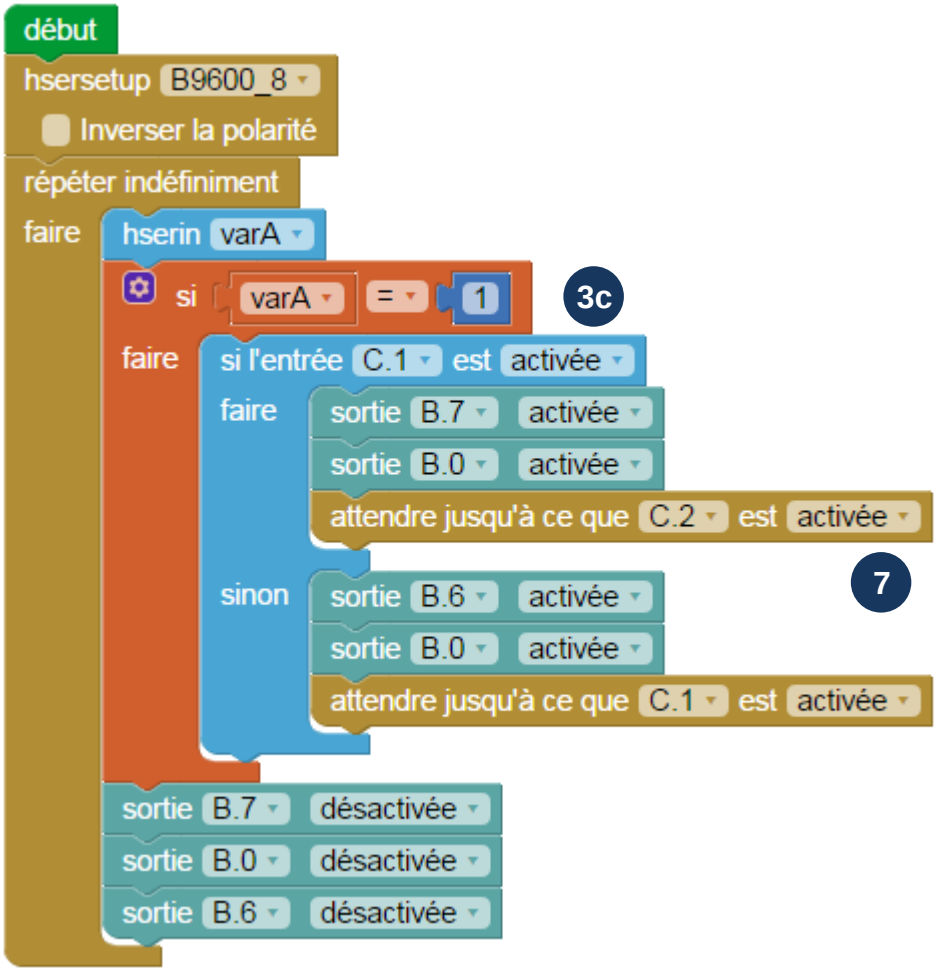
PICAXE **IDE Logicator ou PICAXE Editor 6**



3c Déclenchement de l'ouverture ou de la fermeture.

5 Gestion et attente de réception des données émises par le smartphone.

7 Gestion de l'ouverture et de la fermeture du portail.



3c Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

5 Gestion et attente de réception des données émises par le smartphone.

7 Gestion de l'ouverture et de la fermeture du portail.

Programme Arduino F2_PCOUL.ino

IDE Arduino

(extrait du programme)

```
...
//Programme principal
//Boucle infinie
void loop()
{
 if (Serial.available() > 0) // Y a-t-il de l'activité sur le port série ?
 {
  commande_recue = Serial.read();
  if (commande_recue == 1) // Rentrer dans la boucle si la commande_reçue est "1"
   {
    if (digitalRead(fdc_ouverture)==LOW)
   {
    do
    {
     digitalWrite(ouverture, HIGH); // activation du moteur : sens ouverture
     digitalWrite(gyrophare, HIGH); // déclenchement du gyrophare
    }while (digitalRead(fdc_ouverture)==LOW); // rester dans la boucle tant que fdc_ouverture n'est pas activé

   digitalWrite(ouverture, LOW);
   digitalWrite(fermeture, LOW); // arrêt du moteur
   digitalWrite(gyrophare, LOW); // extinction du gyrophare
  }
   else
   {
    do
     {
       digitalWrite(fermeture, HIGH); // activation du moteur : sens fermeture
       digitalWrite(gyrophare, HIGH); // déclenchement du gyrophare
     } while (digitalRead(fdc_fermeture)==LOW); // rester dans la boucle tant que fdc_fermeture n'est pas
   activé


   digitalWrite(ouverture, LOW);
   digitalWrite(fermeture, LOW); // arrêt du moteur
   digitalWrite(gyrophare, LOW); // extinction du gyrophare
   }
  }
 }
}
// Fin du Programme principal
...
```

3d Déclenchement de l'ouverture ou de la fermeture.

9. Fiche technique N°3- Télécommande ouverture + Sonnette

But de l'application

Etre alerté par un son émis par le smartphone de l'appui du bouton-poussoir extérieur par un visiteur et déclencher l'ouverture du portail à distance.

Notions de programmation abordées

Réception et traitement d'une commande reçue par le smartphone.
Envoi d'une commande du smartphone vers le module Bluetooth du portail.

Fichiers téléchargeables sur www.a4.fr

Application Android	***F3_PCOUL.apk***
Code source App Inventor2	***F3_PCOUL.aia***
Code source Arduino Uno	***F3_PCOUL.ino***
Code source PICAXE Logicator	***F3_PCOUL_LG.plf***
Code source PICAXE Editor 6	***F3_PCOUL_PE6.plf***
Code source PICAXE Logicator pour Autoprog 28X1	***F3_PCOUL_LG_28X1.plf***
Code source PICAXE Editor 6 pour Autoprog 28X1	***F3_PCOUL_PE6_28X1.plf***
Code source PICAXE Blockly	***F3_PCOUL.xml***

9.1. Fonctionnement de l'application

Interface utilisateur sur le Smartphone

Le bouton **Connexion** permet d'établir la connexion entre le smartphone et le module Bluetooth.

Le bouton **Ouvrir portail** déclenche l'ouverture ou la fermeture du portail par envoi du code « 1 ».

Ajout d'une fonction permettant de scruter la réception d'une donnée sur le Smartphone. Si cette donnée correspond à 10, alors le smartphone joue un son de sonnette.

Programme d'automatisme

L'appui sur le bouton-poussoir extérieur ou intérieur envoie le code « 10 » .

La réception du code « 1 » déclenche l'ouverture du portail puis sa fermeture automatique.

Suggestions de modifications

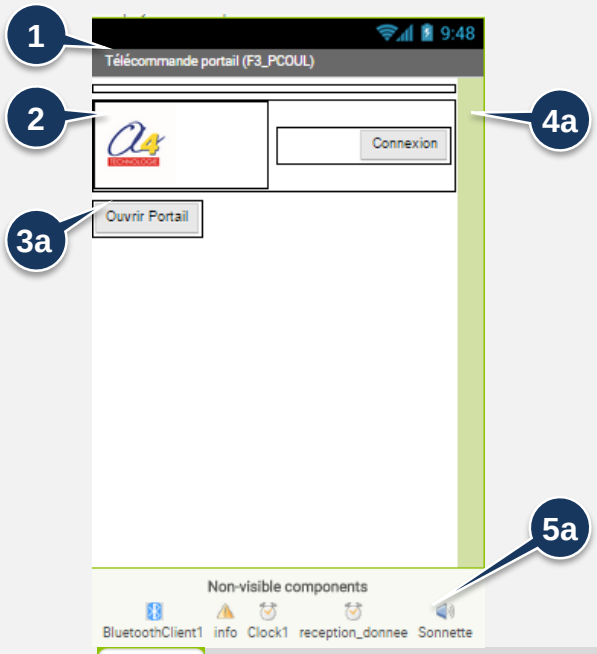
Interface utilisateur
Afficher le texte « Portail fermé » lorsque le portail est fermé.

Programmes
Envoyer un code spécifique dès que le portail est fermé.

Programme App Inventor **F3_PCOUL.aia**

MIT App Inventor | Designer

IDE App Inventor2 (fenêtre Designer)



1 Titre de l'application.

2 Image insérée dans l'application.

3a Déclenchement de l'ouverture ou de la fermeture en envoyant le code **« 1 ».**

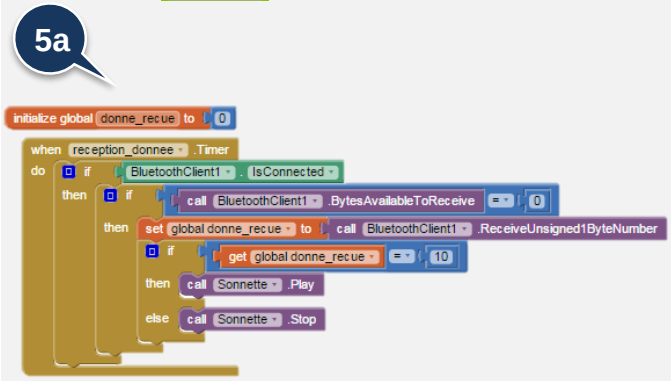
4a Gestion de la communication en Bluetooth.

5a Ajout d'un élément Son
(Disponible dans Palette => Media => Sound)

Ajout d'une Horloge
(Disponible dans Palette => Sensor => Clock)

MIT App Inventor | Blocks

IDE App Inventor2 (fenêtre Blocks)



5a

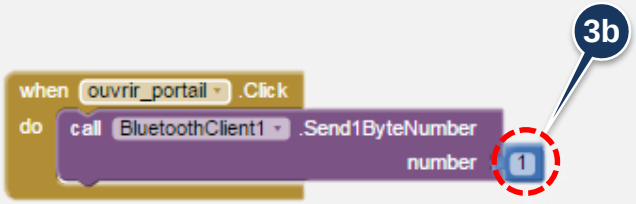
Réception des données

Le bloc **reception_donnee** permet de vérifier cycliquement si une donnée est reçue.
La donnée sera ensuite stockée dans la variable **donnee_recue.**

Si la donnée reçue par le smartphone est égale à **« 10 »** alors jouer l'air de la sonnette**.**

Sinon arrêter la sonnette.

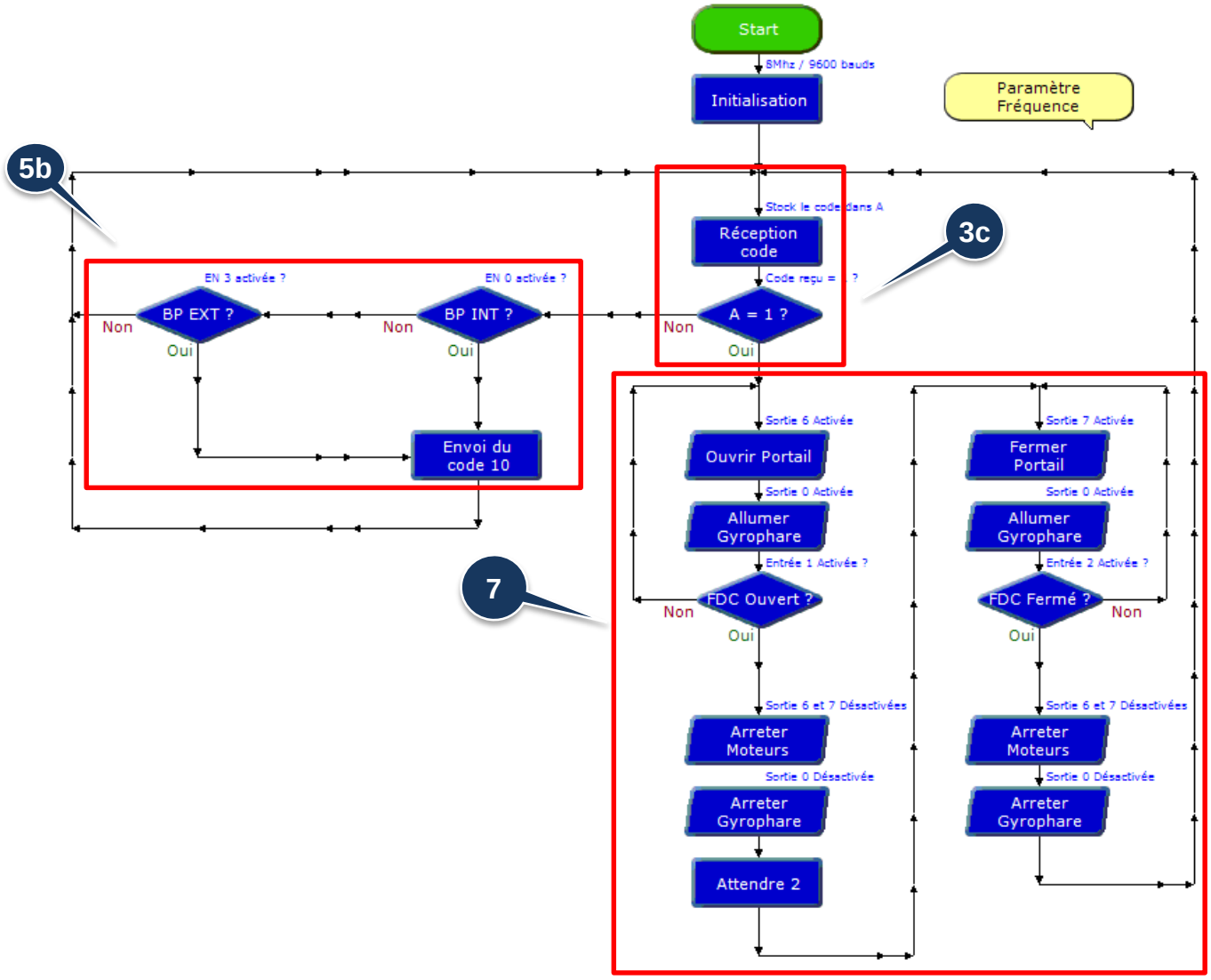
Initialize global donnee_recue to 0 initialise une variable globale nommée « donnee_recue » à **« 0 ».**



3b

Déclenchement de l'ouverture du portail

Lorsque vous cliquez sur le bouton nommé **ouvrir_portail** :
Si le module Bluetooth et le smartphone sont connectés, envoyer le code « 1 » par liaison Bluetooth.
Sinon, afficher « Bluetooth non connecté ».



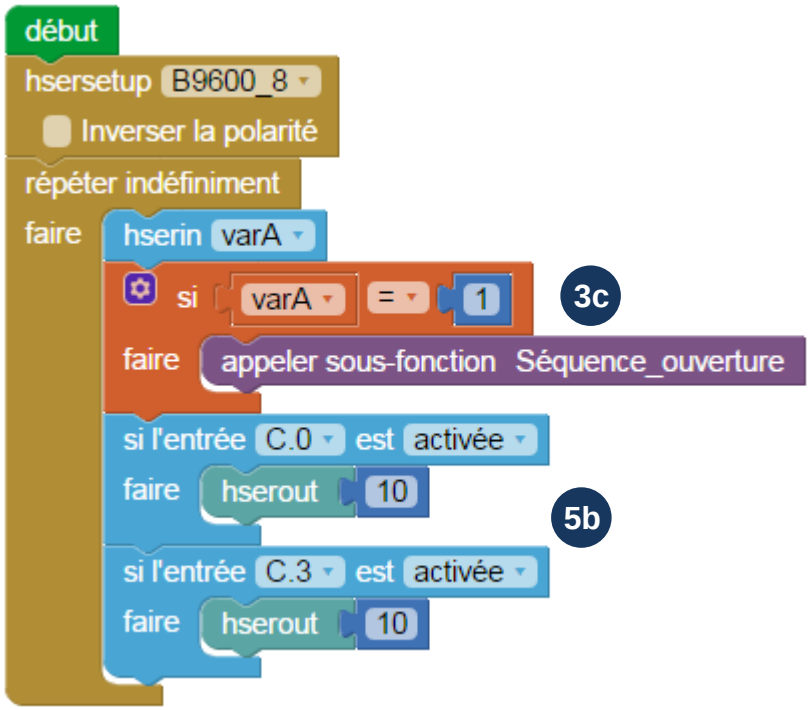
3c Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

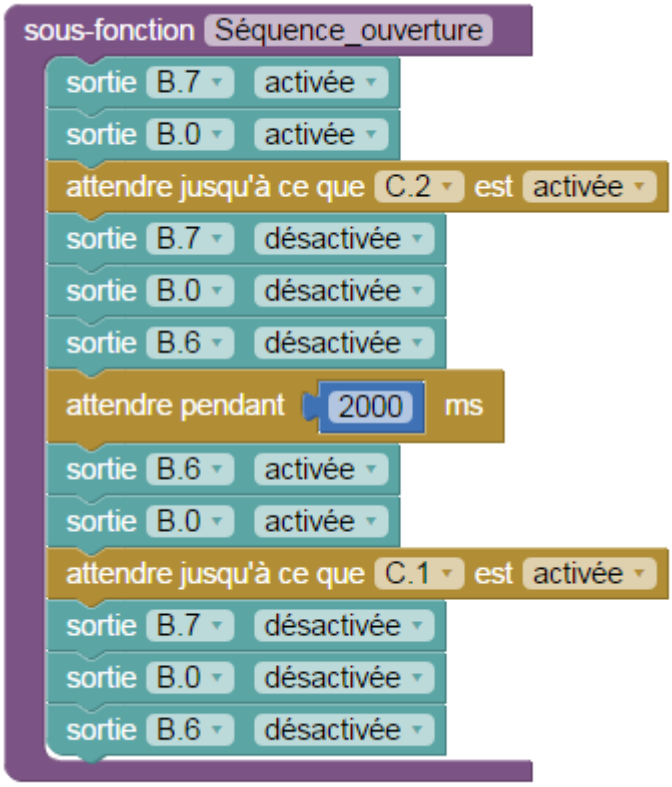
5b Gestion de l'appui sur le bouton-poussoir et l'émission de la donnée (code « 10 ») vers le smartphone.

7 Gestion de l'ouverture et de la fermeture du portail.

Programme Blockly F3_PCOUL.xml

PICAXE **PICAXE Blockly**





3c Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

5b Gestion de l'appui sur le bouton-poussoir et l'émission de la donnée (code « 10 ») vers le smartphone.

Programme Arduino F3_PCOUL.ino

IDE Arduino

(extrait du programme)

```
...
//Programme principal

//Boucle infinie
void loop()
{
if  (digitalRead(bp_int)==HIGH || digitalRead(bp_ext)==HIGH) // Si appui sur bp int ou bp ext
 {
   Serial.write(10); // envoie du code 10 lors de l'appui sur un bouton poussoir
   delay(100);

 }

 if (Serial.available() > 0) //activité sur le port série ?
 {

  commande_recue = Serial.read(); //code reçu


  if (commande_recue == 1)  // Rentrer dans la boucle si la commande_reçue est "1"
  {
    Serial.write(1); // envoie du code 1 pour couper la sonnerie

     do
     {
      digitalWrite(ouverture, HIGH); // activation du moteur : sens ouverture
      digitalWrite(gyrophare, HIGH); // déclenchement du gyrophare
     }while (digitalRead(fdc_ouverture)==LOW); // rester dans la boucle tant que fdc_ouverture n'est pas activé

    digitalWrite(ouverture, LOW);
    digitalWrite(fermeture, LOW); // arrêt du moteur
    digitalWrite(gyrophare, LOW); // extinction du gyrophare

   delay (2000); // delai de 2 secondes entre l'ouverture et la fermeture

     do
      {
        digitalWrite(fermeture, HIGH); // activation du moteur : sens ouverture
        digitalWrite(gyrophare, HIGH); // allumage du gyrophare
      } while (digitalRead(fdc_fermeture)==LOW); // rester dans la boucle tant que fdc_fermeture n'est pas activé

    digitalWrite(ouverture, LOW);
    digitalWrite(fermeture, LOW); // arrêt du moteur
    digitalWrite(gyrophare, LOW); // extinction du gyrophare
    }

  }

}
  // Fin du Programme principal// Fin du Programme principal
...
```

3d Vérification du code reçu pour la demande d'ouverture du portail (code « 1 »).

5c Gestion de l'appui sur le bouton-poussoir et l'émission de la donnée (code « 10 ») vers le smartphone.

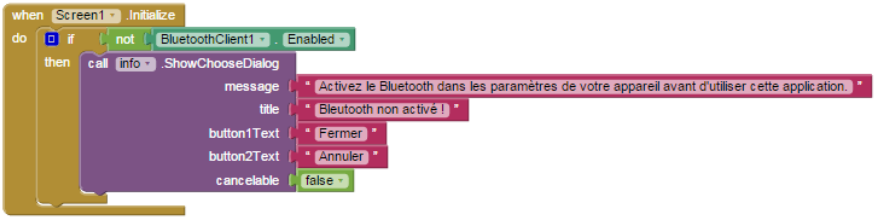
10. ANNEXE

10.1. Gestion de l'émission de données vers l'interface programmable

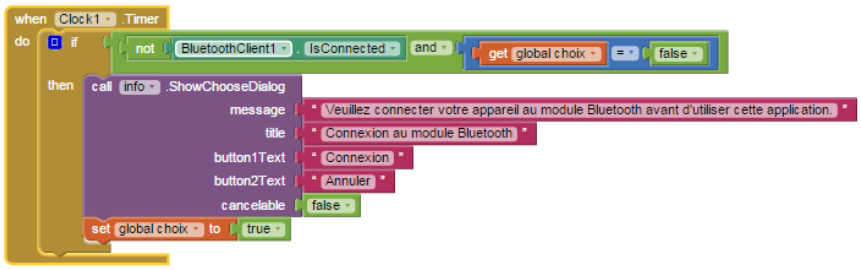
Les blocs suivants sont nécessaires pour initialiser la communication de données en Bluetooth du smartphone vers l'interface programmable.

Attention : Ces blocs sont indispensables à la communication entre le smartphone et le module Bluetooth. Ils doivent être modifiés avec précaution.

Gestion d'un message d'alerte : Bluetooth désactivé-non connecté



Affichage d'une alerte sur l'écran pour indiquer que le Bluetooth est désactivé dans les paramètres du smartphone.

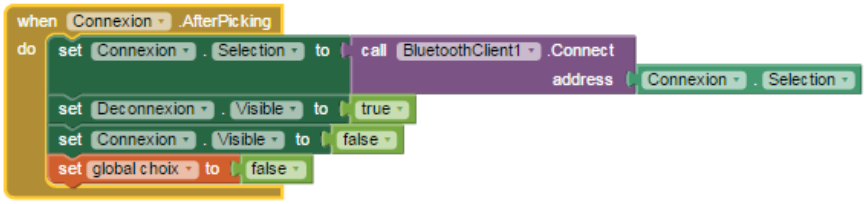


Affichage d'un message demandant de se connecter au module Bluetooth. Un message d'alerte apparaitra cycliquement si les deux appareils ne sont pas connectés.

Affichage des appareils appairés

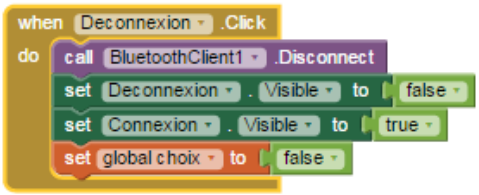
Affichage de la liste des appareils appairés avec le smartphone.

Sélection de l'appareil à connecter au smartphone



Établissement de la connexion avec l'appareil sélectionné et affichage du bouton **Déconnexion**.

Gestion de la déconnexion

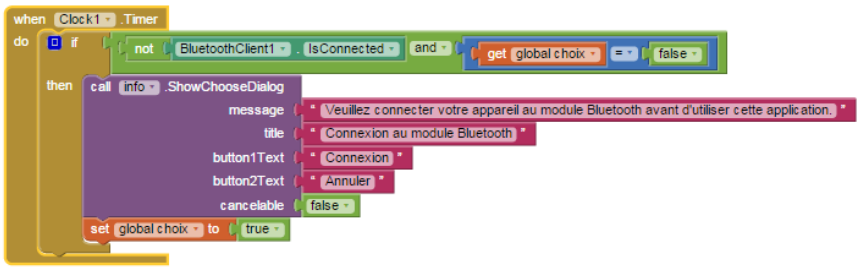


Déconnexion de l'appareil et affichage du bouton **Connexion**.

10.2. Gestion de la réception de données en provenance de l'interface programmable

Les blocs suivants sont nécessaires pour initialiser la réception de données en Bluetooth.

Traitement d'une donnée reçue



Le smartphone vérifie cycliquement que le module Bluetooth est connecté.

Si le smartphone n'est pas connecté au module Bluetooth, un message d'alerte apparaît avec la possiblité de se connecter directement à partir de ce message.

Bloc BASIC Réception Code sous PICAXE

```
if hserflag = 1 then
        do
                let varA=@ptrinc
        loop while ptr<>hserptr
        let hserflag = 0
else let varA=0

end if
```

Si une donnée est reçue

Affecter la derniere valeur obtenue à la variable A

Si non réinitialiser la variable A à 0

Fin de Si

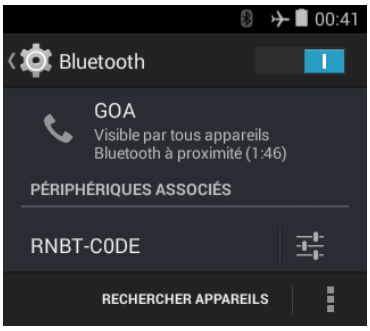
10.3. Mise en service du module Bluetooth

1

Configurer le module Bluetooth comme indiqué au chapitre 3 et le connecter à l'interface AutoProg ou AutoProgUno en respectant le plan de câblage proposé au chapitre 4.

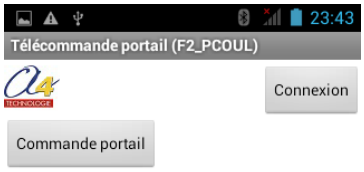
À la mise sous tension du module (au travers de l'interface AutoProg ou AutoProgUno), le témoin rouge **PWR** est allumé et le témoin vert **ETAT** clignote.

2



Activer le Bluetooth et cliquer sur **Rechercher Appareils**.
Sélectionner le module Bluetooth du portail coulissant pour procéder à l'appairage.

3



Lancer l'application puis appuyer sur le bouton **Connexion** en haut à droite de votre écran.

4

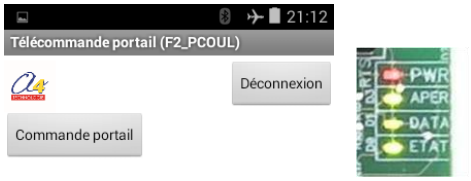
Sélectionner l'identifiant du module Bluetooth auquel il faut connecter le smartphone.
L'identifiant correspond à l'inscription **MAC ID** que l'on peut lire sur le composant Bluetooth qui équipe le module.

Le témoin vert **ETAT** cesse de clignoter et reste allumé en permanence.
Le témoin vert **APER** indique que la connexion entre le smartphone et le module Bluetooth est effectuée.

4bis

Si ce message d'erreur apparaît, vérifier que le module est sous tension.
Quitter éventuellement l'application, appuyer pendant 5 s sur le bouton **RESET** de la carte Bluetooth, redémarrer le smartphone.
Répéter l'opération de connexion à partir de l'étape N°3.

5



Lorsque la connexion est réalisée, le bouton **Déconnexion** apparaît.
Le témoin vert **DATA** s'allume dès qu'une donnée est émise ou reçue par le module Bluetooth.
L'appui sur le bouton d'envoi de données, dans cet exemple **Commande portail**, déclenche l'allumage fugitif de ce témoin.

Édité par la société A4 Technologie
Tél. 01 64 86 41 00 - Fax : 01 64 46 31 19
www.a4.fr