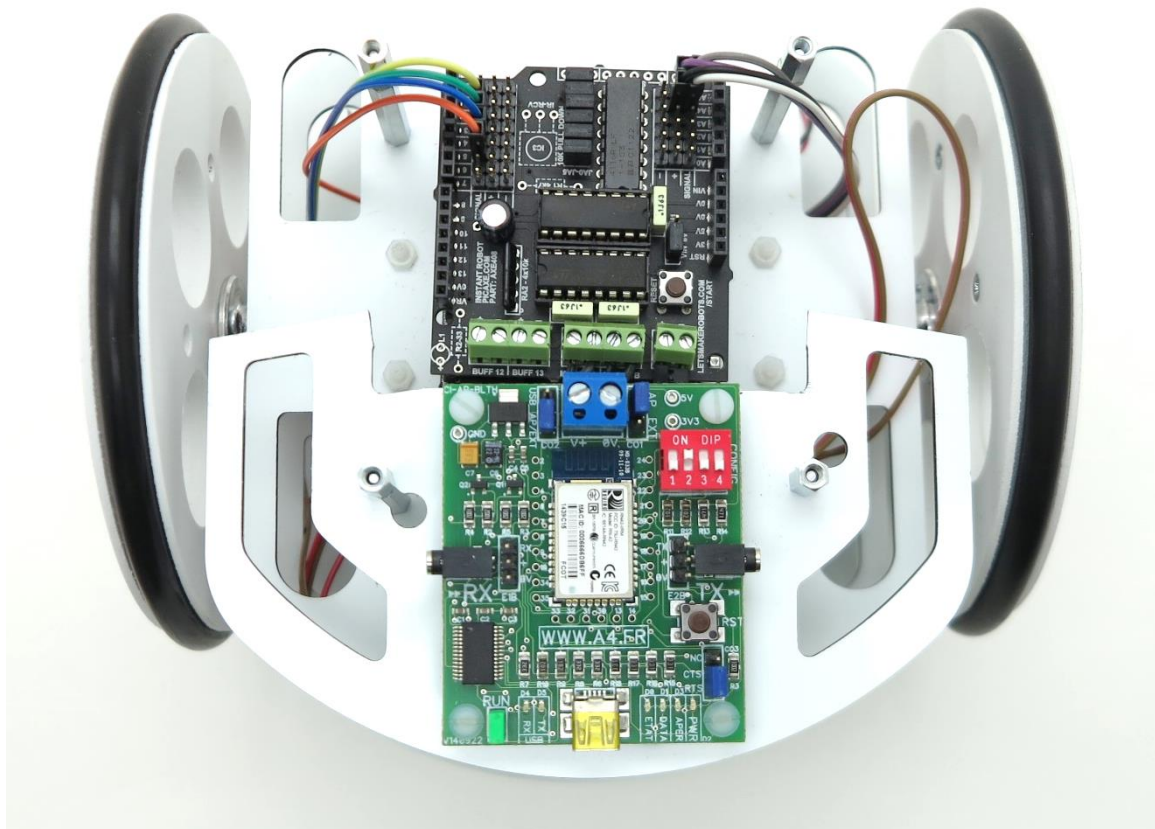


RoboCoda

Plateforme robotique modulaire

**Option Bluetooth
avec Arduino et ApplInventor**





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.



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

La duplication de ce dossier est autorisée sans limite de quantité au sein des établissements scolaires, aux seules fins pédagogiques, à la condition que soit cité le nom de l'éditeur : Sté A4.

La copie ou la diffusion par quelque moyen que ce soit à des fins commerciales n'est pas autorisée sans l'accord de la Sté A4.



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



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

SOMMAIRE

1. Introduction.....	2
1.1. Prérequis souhaitables	2
1.2. Organisation de ce dossier	2
2. Eléments nécessaires	3
2.1. Matériels	3
2.2. Logiciels.....	3
2.3. Ressources complémentaires	3
3. Le module Bluetooth	4
3.1. Configuration	4
3.2. Témoins lumineux.....	4
4. Tableau d'affectation des Entrées/Sorties Arduino.....	5
5. Mise en service des applications.....	6
5.1. Programmes pour smartphone Android	6
6. Fiche technique N°1 – RC_B1_Avancer	8
7. Fiche technique N°2 – RC_B2_Pilotage	11
8. Fiche technique N°3 – RC_B3_Accelerometre.....	14
Fiche technique N°4 – RC_B4_Reception_donnee	17
9. Fiche technique N°2 – RC_B5_Choix_pilotage.....	20

1. Introduction

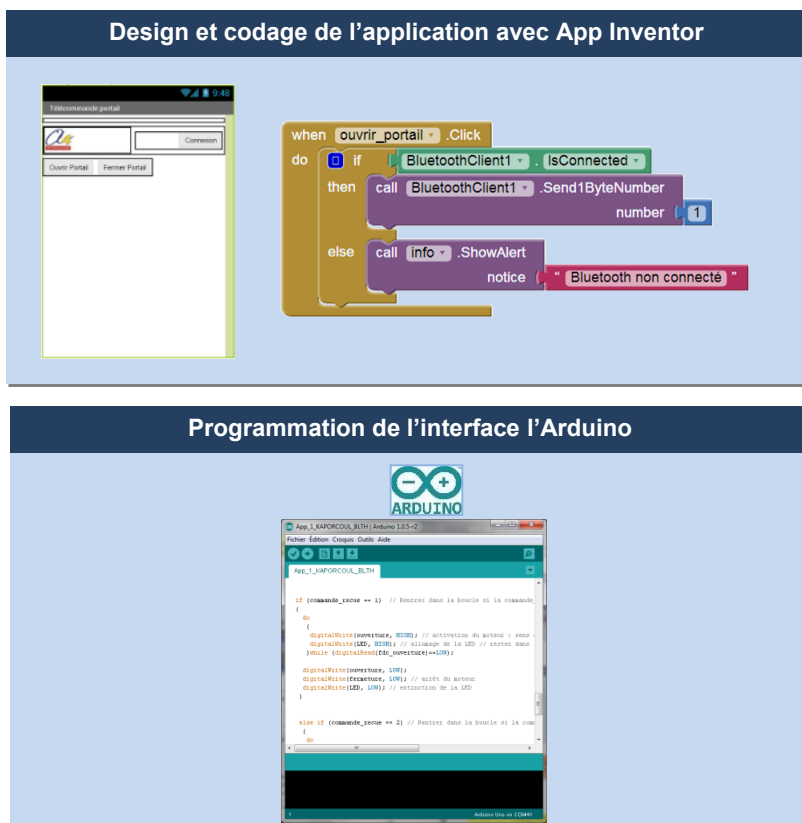
Ce document illustre l'utilisation du module Bluetooth au travers d'exemples basés sur **le robot CoDa d'A4** équipée avec **l'option module Bluetooth**.

Les applications proposées permettent d'interagir entre une application chargée sur **Smartphone Android** et la maquette pilotée par une interface programmable **Arduino Uno**.

Les programmes sont développés avec les IDE* :

- **App Inventor 2** pour les applications Android ;
- **IDE Arduino** pour les applications robotique ;

* IDE : Environnement de Développement Intégré



Programmation Smartphone



Bluetooth®

Programmation Interface



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 les programmes de bases du robot CoDa avant de s'initier avec l'option Bluetooth

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 robot.

Des propositions de modifications sont suggérées à la fin de chaque fiche.

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. Eléments nécessaires

2.1. Matériels

- **Robot CoDa** monté avec la carte **Arduino Uno**.
- **Option Bluetooth** montée sur robot **CoDa**.
- **Câble de programmation USB type B** pour la programmation de la carte Arduino.

2.2. Logiciels

- IDE Arduino (<http://arduino.cc/en/Main/Software>) pour la programmation de l'interface Arduino Uno.
- 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.

* IDE : *Environnement de Développement Intégré*

2.3. Ressources complémentaires

Des ressources complémentaires sont disponibles sur www.a4.fr

- Dossier Technique Robot CoDa
- Dossier App Inventor 2 pour prendre en main AppInventor 2
- Guide d'utilisation PICAXE Logicator

D'autres ressources sont également 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.

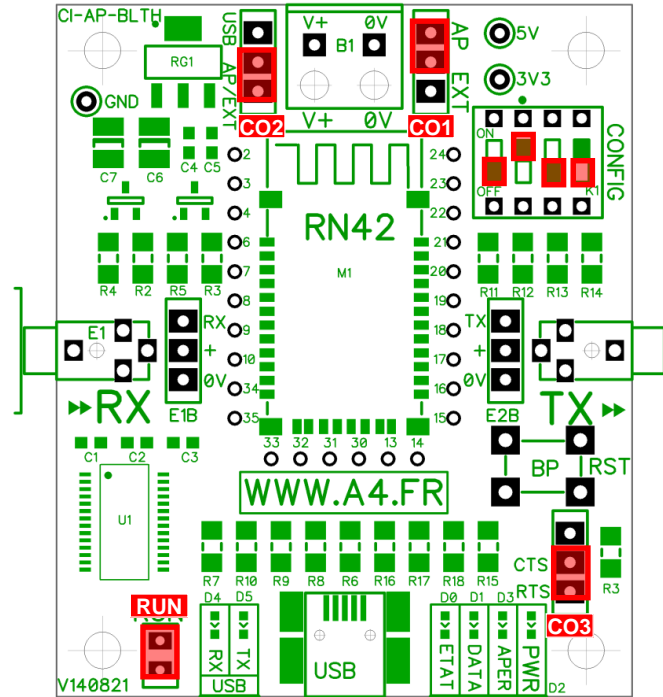
En mode avancé, il peut être configuré au travers d'une liaison par connexion USB à un PC ou par l'envoi de commandes au travers de ses liaisons RX et TX.

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. Tableau d'affectation des Entrées/Sorties Arduino

Ci-dessus le tableau résumant la connexion entre les broches de la carte Shield AXE408 et la carte de commande Arduino Uno. Ce tableau sert principalement à faire le lien entre le câblage des capteurs/actionneurs et la programmation du microcontrôleur.

N° Broche AXE 408 (Shield moteur)	MODULE CAPTEUR/ACTIONNEURS	Variable Arduino associée
0 (RX)	Module Bluetooth (TX)	commande_recue
1 (TX)	Module Bluetooth (RX)	C.6
2	Détecteur de ligne Droit	detecteur_droit
3	Détecteur de ligne Central	detecteur_central
4	Détecteur de ligne Gauche	detecteur_gauche
5	Module capteur à ultrasons	ultrasons
6	Broche libre	B.6
7	Broche libre	B.7
8	Direction moteur droit	direction_moteur_droit
9	Puissance moteur droit	puissance_moteur_droit
10	Puissance moteur gauche	puissance_moteur_gauche
11	Direction moteur gauche	direction_moteur_gauche
12	Broche libre	C.4
13	Broche libre	C.3
A0	Broche libre	A.0
A1	Broche libre	A.1
A2	Broche libre	A.2
A3	Broche libre	A.3
A4	Microrupteur gauche	contact_gauche
A5	Microrupteur droit	contact_droit

5. Mise en service des applications

Les applications proposées dans les fiches suivantes sont créées avec différents IDE (Environnement de Développement Intégré).

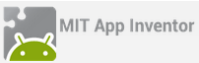
Pour chaque application, un ensemble de fichiers à charger dans le smartphone Android et dans l'interface programmable Arduino est proposé.


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.


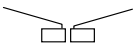


5.1. Programmes pour smartphone Android

	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	<ul style="list-style-type: none">- 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.





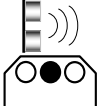
	http://arduino.cc/en/Main/Software (rubrique Arduino IDE) Cette IDE est gratuite. Elle doit être installée sur un PC.
Matériel associé	<ul style="list-style-type: none">- Interface programmable AutoProgUno ou autre carte Arduino compatible.- Cordon de liaison USB type imprimante avec le PC pour transférer les programmes dans l'interface AutoProgUno.
Fichiers	Fichiers .INO à ouvrir dans l'IDE Arduino et à téléverser dans l'interface.

Liste des symboles utilisés pour les capteurs:

(Voir le chapitre Montage / Assemblage pour la mise en œuvre des capteurs)

Symbole	Description
	Aucun capteur
	Module capteurs microrupteurs (détection de contact avec un obstacle).
	Module capteurs infrarouges (détection de marquage au sol).
	Module capteur à ultrasons (détection d'obstacle à distance).

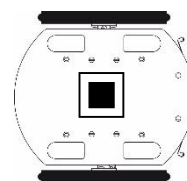
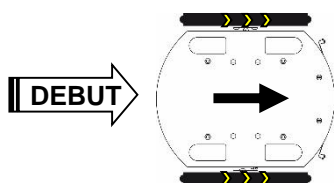
Liste des programmes

Fiche	Capteur	Description	Fichier programme
N°1		Envoyer une instruction du Smartphone au robot	RC_B1_Avancer
N°2		Piloter le robot avec le Smartphone	RC_B2_Pilotage
N°3		Piloter le robot avec l'accéléromètre du Smartphone	RC_B3_Accelerometre
N°4		Recevoir une donnée émise par le module Bluetooth	RC_B4_Reception_donnee
N°5		Etablir un mode de fonctionnement manuel et automatique	RC_B5_Choix_pilotage

6. Fiche technique N°1 – RC_B1_Avancer

But de l'application	Se déplacer en avant et s'arrêter avec les commandes du Smartphone
Notions de programmation abordées	Pilotage du robot sans fil par connexion Bluetooth à un appareil Android
Programme associé	Code source Arduino RC_B1_Avancer.ino Code source App Inventor RC_B1_Avancer.aia Application Android RC_B1_Avancer.apk (Fichiers téléchargeables sur www.a4.fr)
Capteur mis en jeu	Aucun

Illustration



Suggestions de modifications	Modifier le programme pour que le robot avance lorsque le bouton avancer est appuyé et s'arrête au relâchement
------------------------------	--

MIT App Inventor Designer IDE App Inventor2 (fenêtre Designer)

1 Titre de l'application.

2 Image insérée dans l'application.

3a Avance du robot en envoyant le code « 1 ».

4a Arrêt du robot 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.

MIT App Inventor Blocks IDE App Inventor2 (fenêtre Blocks)

3b **Déclenchement de la commande avancer**
Lorsque bouton nommé **avancer** est cliqué :
Le smartphone envoie le code « 1 » par liaison Bluetooth.

4b **Déclenchement de la commande arret**
Lorsque bouton nommé **arret** est cliqué :
Le smartphone envoie 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.

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 ».



La variable commande reçue servira de mémoire pour stocker la valeur envoyée par le Smartphone et reçue par le module Bluetooth :

```
int commande_recue;
```

L'ouverture du port Série est nécessaire pour activer la communication entre le l'Arduino Uno et le module Bluetooth à 9600 bauds.

```
Serial.begin(9600); // Initialisation de la liaison Série
```

3c Si-dessous le code permettant de vérifier si un code est bien reçu. Les opérations sont effectuées en fonction du code reçu.

```
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_recue est '1'
        {
            avancer();
        }

        if (commande_recue == 2) // Rentrer dans la boucle si la commande_recue est '2'
        {
            arret();
        }
    }
}
```

3c Vérification du code reçu (code « 1 »).

4c Vérification du code reçu (code « 2 »).

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

7. Fiche technique N°2 – RC_B2_Pilotage

But de l'application	Piloter le robot dans toutes les directions. Le robot doit se déplacer seulement sur l'appui du bouton et doit s'arrêter au relâchement.
Notions de programmation abordées	Contrôler la direction du Robot, nouvelle instruction sur le déclenchement d'une séquence App Inventor
Programme associé	Code source Arduino RC_B2_Pilotage.ino Code source App Inventor RC_B2_Pilotage.aia Application Android RC_B2_Pilotage.apk (Fichiers téléchargeables sur www.a4.fr)
Capteur mis en jeu	Aucun
Suggestions de modifications	Ajouter un contrôle de la vitesse des moteurs directement disponible sur l'appareil Android.

1 Titre de l'application.

2 Image insérée dans l'application.

3a Avance du robot lors de l'appui du bouton en envoyant le code « 1 ».

4a Arrêt du robot lors du relâchement du bouton en envoyant le code « 0 ».

5a Gestion de la communication en Bluetooth.

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

3b Avancer :
Lorsque le bouton nommé avancer est enfoncé :
 Envoyer le code « 1 » par liaison Bluetooth.
 Cette instruction est répétée pour les 3 autres commandes de pilotage du robot avec un code envoyé différent pour chaque instruction.

4b Arrêt du robot :
Lorsque le bouton nommé avancer est relâché
 envoyer le code « 0 » par liaison Bluetooth.
 Cette instruction est répétée pour les 3 autres commandes de pilotage avec le code « 0 »

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.

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 ».



La vérification du code reçu par le module Bluetooth et transmis par le smartphone se fait par l'instruction **Switch Case**. Le test est fait sur la variable **commande_recue**

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

    switch (commande_recue)
    {

      case 0 : // Si la commande recue vaut '0'
        arret();
        break;

      case 1 : // Si la commande recue vaut '1'
        avancer();
        break;

      case 2 : // Si la commande recue vaut '2'
        gauche();
        break;

      case 3 : // Si la commande recue vaut '3'
        droite();
        break;

      case 4 : // Si la commande recue vaut '4'
        reculer();
        break;

      default :// Si une commande non connue est recue.
        arret();
    }
  }
}
```

8. Fiche technique N°3 – RC_B3_Accelerometre

But de l'application	Piloter le robot dans toutes les directions en fonction de l'inclinaison de l'appareil
Notions de programmation abordées	Contrôler la direction du Robot à l'aide du capteur Accéléromètre
Programme associé	Code source Arduino RC_B3_Accelerometre.ino Code source App Inventor RC_B3_Accelerometre.aia Application Android RC_B3_Accelerometre.apk (Fichiers téléchargeables sur www.a4.fr)
Capteur mis en jeu	Aucun
Suggestions de modifications	Gérer la vitesse des moteurs en fonctions de l'inclinaison de l'appareil.

IDE App Inventor2 (fenêtre Designer)

- 1 Titre de l'application.
- 2 Image insérée dans l'application.
- 3a Variable d'inclinaison de l'appareil. X, Y et Z afficheront l'inclinaison de chaque axe de l'appareil.
- 4a Capteur accéléromètre
- 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 Affichage des variables :
Lorsque l'inclinaison de l'appareil change, les variables X, Y et Z prennent les valeurs d'inclinaison.

4b Envoi du code :
Dans cette partie le code est envoyé en permanence, sans action nécessaire sur un bouton.
Lorsque les valeurs de l'accéléromètre changent, l'appareil Android envoie automatiquement un code au module 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.

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 ».



De même que le projet précédant la vérification du code reçu par le module Bluetooth et transmis par le smartphone se fait par l'instruction **Switch Case**. Le test est fait sur la variable **commande_recue**

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

    switch (commande_recue)
    {

      case 0 : // Si la commande recue vaut '0'
        arret();
        break;

      case 1 : // Si la commande recue vaut '1'
        avancer();
        break;


      case 2 : // Si la commande recue vaut '2'
        gauche();
        break;

      case 3 : // Si la commande recue vaut '3'
        droite();
        break;

      case 4 : // Si la commande recue vaut '4'
        reculer();
        break;

      default :// Si une commande non connue est recue.
        arret();
    }
  }
}
```

Fiche technique N°4 – RC_B4_Reception_donnee

But de l'application	Réceptionner des données reçues par les capteurs du robot.
Notions de programmation abordées	Emission de données sur Arduino, réception de données sur App Inventor.
Programme associé	Code source Arduino RC_B4_Reception_donnee.ino Code source App Inventor RC_B4_Reception_donnee.aia Application Android RC_B4_Reception_donnee.apk (Fichiers téléchargeables sur www.a4.fr)
Capteur mis en jeu	 Capteur à Ultrasons
Suggestions de modifications	Afficher un texte sur l'application lorsque la distance reçue est inférieure à 10 cm.

1 Titre de l'application.

2 Image insérée dans l'application.

3a Variable qui stockera la valeur envoyée par le module Bluetooth.

4a Insertion d'une nouvelle horloge pour vérifier la réception de donnée sur l'appareil.

5a Gestion de la communication en Bluetooth.

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

3b Affichage de la donnée

La donnée reçue est stockée et est affichée dans la variable **distance_US**.

4b Initialisation d'une nouvelle horloge

A chaque coup d'horloge, si le Bluetooth est connecté, l'appareil scrute pour voir si une donnée a été émise par le module Bluetooth.

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.

5b

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 ».



Initialisation d'une variable qui stockera la distance perçue par le module à Ultrasons :

```
int distance = 0;
```

La distance est envoyée par liaison série au module Bluetooth qui la transmettra ensuite au Smartphone avec l'instruction ***Serial.write()***;

```
void loop()
{
    distance = mesure_distance(); // aller dans la sous-fonction mesure_distance et renvoyer la valeur mesurée
    //par le capteur ultrason dans la variable distance
    Serial.write(distance); // envoie la valeur de la distance au module Bluetooth
    delay (100); // delai de traitement obligatoire 100 ms
}
```


3c

Envoi la valeur de la variable B à l'appareil Android

5c

Initialisation de la carte Arduino pour l'envoi des données

9. Fiche technique N°2 – RC_B5_Choix_pilotage

But de l'application	Fusionner deux modes de pilotage : <ul style="list-style-type: none">- Un pilotage Automatique qui entrainera le robot à suivre une ligne.- Un pilotage manuel, où l'utilisateur reprend le contrôle du robot. Inclure une sécurité. Stopper le robot lorsque le capteur à Ultrasons détecte un obstacle.
Notions de programmation abordées	Rendre compatible plusieurs modes de pilotages. Gérer l'envoi et la réception de donnée en simultanée
Programme associé	Code source Arduino RC_B5_Choix_pilotage.ino Code source App Inventor RC_B5_Choix_pilotage.aia Application Android RC_B5_Choix_pilotage.apk (Fichiers téléchargeables sur www.a4.fr)
Capteur mis en jeu	 Capteur à Ultrasons et Module capteurs infrarouges

1 Titre de l'application.

2a Application **RC_B2_Pilotage**.

3a Ajout de deux boutons qui détermineront le choix du pilotage.

4a Application **RC_B4_Reception_donnee**.

5a Gestion de la communication en Bluetooth.

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

3b

Choix du mode de pilotage :

Lorsque le bouton pilotage_auto est appuyé, le Smartphone envoie le code « **254** ».

Lorsque le bouton pilotage_manuel est appuyé, le Smartphone envoie le code « **255** ».

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 ».



Une variable nommée **choix_programme** est déclarée et servira de condition de pilotage. Cette variable déterminera dans quel type de pilotage le robot doit suivre.

```
void loop()
{

    if (choix_programme == 1)
    {
        pilotage_manuel();
    }

    else if (choix_programme == 0)
    {
        pilotage_auto();
    }
}
```

La gestion des deux modes est ensuite gérée séparément. Il est nécessaire de vérifier à chaque fin de fonction si un changement de mode de pilotage a lieu pour modifier la variable **choix_programme**.