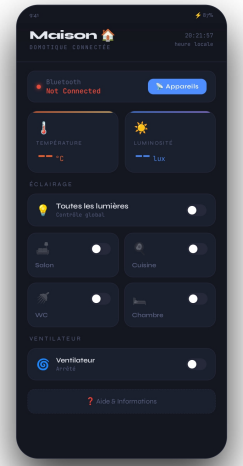


Projet guidé - MIT App Inventor

Maison

Domotique connectée

Construire une application Android qui contrôle une mini-maison connectée par Bluetooth : capteurs, lumières et ventilateur.



Version	Finale - sans section Modes
Public	Élèves francophones
Durée	2 à 4 séances
Matériel	Android + App Inventor + Arduino + HC-05/HC-06

Résultat attendu
Une application sombre et lisible avec bouton Bluetooth, température, luminosité, éclairage global, quatre zones de lumière, ventilateur et aide.

Sommaire de construction

Ce document est un mode d’emploi complet. Les élèves doivent suivre l’ordre : interface, noms des composants, blocs Bluetooth, commandes, réception des capteurs, test final.

- Cahier des charges et fonctions obligatoires
- Matériel et préparation
- Architecture de l’écran
- Construction précise dans Designer
- Liste officielle des composants
- Procédure SendCommand(cmd)
- Connexion Bluetooth
- Interrupteurs et alphabet de commandes
- Réception température / luminosité
- Code Arduino de référence
- Tests et dépannage
- Barème de validation

Règle de travail
Ne pas changer les noms des composants pendant le tutoriel. Les blocs dépendent directement de ces noms.

1. Cahier des charges

L’application doit reproduire un module unique de maison connectée. Les fonctions sont limitées volontairement pour rester fiables : pas de sous-écrans, pas de modes supplémentaires, pas de fonctions décoratives inutiles.

Fonctions à obtenir

- Un en-tête avec le titre Maison, le sous-titre DOMOTIQUE CONNECTÉE et l’heure locale.
- Une carte Bluetooth avec le statut Not Connected puis Connected.
- Deux cartes capteurs : Température et Luminosité.
- Une section Éclairage avec un contrôle global et quatre zones : Salon, Cuisine, WC, Chambre.
- Une section Ventilateur avec un interrupteur et un état textuel.
- Un bouton Aide & Informations affichant les commandes Bluetooth.

Alphabet Bluetooth officiel

Commande	ON	OFF	Rôle
Toutes les lumières	t	T	Allumer / éteindre toutes les LED
Salon	s	S	Piloter la lumière du salon
Cuisine	c	C	Piloter la lumière de la cuisine
WC	h	H	Piloter la lumière WC
Chambre	w	W	Piloter la lumière chambre
Ventilateur	v	V	Démarrer / arrêter le ventilateur

À ne pas ajouter
La version finale ne contient pas de section saisonnière. Les boutons supplémentaires ont été supprimés pour faciliter le montage, le test et la correction.

2. Matériel et préparation

L’élève doit pouvoir tester sur téléphone Android réel. Le Bluetooth classique HC-05 / HC-06 ne se teste pas correctement dans le navigateur seul.

Élément	Utilisation
Compte MIT App Inventor	Créer et éditer le projet
Téléphone Android	Tester l’application avec AI Companion ou APK
Module HC-05 ou HC-06	Communication Bluetooth classique
Arduino Uno/Nano ou équivalent	Recevoir les commandes et envoyer les capteurs
LED ou relais	Simuler / piloter les lumières
DHT11/DHT22 ou capteur température	Fournir la température
Photorésistance ou module luminosité	Fournir la luminosité

Préparation avant App Inventor

1. Appairer le téléphone avec le module Bluetooth dans les réglages Android.
2. Noter le nom du module : souvent HC-05 ou HC-06.
3. Préparer le code Arduino ou au minimum un programme qui répond aux lettres reçues.
4. Créer un nouveau projet nommé Maison_Connectee dans MIT App Inventor.

Attention Android
Sur certains téléphones récents, il faut autoriser Bluetooth / Nearby devices pour que la liste des appareils s’affiche.

3. Architecture de l’écran

On construit un seul écran vertical et scrollable. Les zones suivent exactement l’ordre du visuel final : header, Bluetooth, capteurs, éclairage, ventilateur, aide.

Ordre	Bloc visuel	Composants principaux
1	En-tête	HorizontalArrangement_Header + labels titre / heure
2	Bluetooth	HorizontalArrangement_BluetoothCard + ListPickerBluetooth
3	Capteurs	HorizontalArrangement_Sensors + deux cartes
4	Éclairage	SwitchAll + TableArrangement_Lights
5	Ventilateur	HorizontalArrangement_Fan + SwitchFan
6	Aide	ButtonHelp + Notifier1

Palette graphique recommandée

Usage	Couleur
Fond principal	#0D0F14
Cartes	#171B27 ou #1A1E2B
Texte principal	#E8EAF0
Texte secondaire	#747D98
Accent bleu	#4F8EFF
Accent orange	#FF6B35
Rouge non connecté	#E74C3C
Vert connecté	#2ECC71

4. Construction dans Designer

Dans Screen1, commencer par régler les propriétés générales, puis ajouter les arrangements dans l’ordre. Le design exact d’App Inventor sera plus simple que le visuel, mais la structure doit être fidèle.

Propriétés de Screen1

```
Screen1.Title = Maison
Screen1.Scrollable = true
Screen1.BackgroundColor = #0D0F14
Screen1.AlignHorizontal = Center
Screen1.Sizing = Responsive
Screen1.Icon = maison_conectee_icon.png
```

Conteneur principal

```
Ajouter VerticalArrangement_Main
Width = Fill parent
Height = Automatic
BackgroundColor = #0D0F14
AlignHorizontal = Center
```

En-tête

- Ajouter HorizontalArrangement_Header.
- À gauche : LabelTitle = Maison et LabelSubtitle = D O M O T I Q U E C O N N E C T É E.
- À droite : LabelTime et LabelTimeSub = heure locale.

Carte Bluetooth

- Ajouter HorizontalArrangement_BluetoothCard.
- Ajouter LabelBluetoothDot avec le texte ●.
- Ajouter LabelBluetooth = Bluetooth et LabelStatus = Not Connected.
- Ajouter ListPickerBluetooth avec le texte Appareils.

4 bis. Cartes et sections restantes

Cartes capteurs

- Ajouter HorizontalArrangement_Sensors.
- Créer VerticalArrangement_TempCard avec LabelTempIcon, LabelTempTitle, LabelTempValue.
- Créer VerticalArrangement_LumCard avec LabelLumIcon, LabelLumTitle, LabelLumValue.
- Valeurs de départ : -- °C et -- lux.

Section éclairage

- Ajouter LabelSectionLights = É C L A I R A G E.
- Ajouter HorizontalArrangement_AllLights avec SwitchAll.
- Ajouter TableArrangement_Lights en 2 colonnes et 2 lignes.
- Créer les quatre cartes : Salon, Cuisine, WC, Chambre, chacune avec une icône, un label et un switch.

Section ventilateur et aide

- Ajouter LabelSectionFan = V E N T I L A T E U R.
- Ajouter HorizontalArrangement_Fan avec LabelFanIcon, LabelFanName, LabelFanStatus, SwitchFan.
- Ajouter ButtonHelp = □ Aide & Informations.

Logo

Dans les propriétés de Screen1, téléverser le fichier maison_conectee_icon.png et l’utiliser comme icône de l’application.

5. Noms officiels des composants

Copier exactement ces noms. Cela évite les erreurs lorsque les blocs seront construits.

Zone	Composants visibles
En-tête	LabelTitle, LabelSubtitle, LabelTime, LabelTimeSub
Bluetooth	LabelBluetoothDot, LabelBluetooth, LabelStatus, ListPickerBluetooth
Capteurs	LabelTempValue, LabelLumValue
Éclairage global	SwitchAll
Éclairage zones	SwitchSalon, SwitchCuisine, SwitchWC, SwitchChambre
Ventilateur	SwitchFan, LabelFanStatus
Aide	ButtonHelp

Composants non visibles

Composant	Nom	Réglage
BluetoothClient	BluetoothClient1	DelimiterByte = 10
Clock	ClockClock	TimerInterval = 1000, TimerEnabled = true
Clock	ClockReceive	TimerInterval = 500, TimerEnabled = true
Notifier	Notifier1	Aucun réglage spécial

Pourquoi DelimiterByte = 10 ?
Parce que le code Arduino envoie les données avec Serial.println(). Le caractère 10 marque la fin de ligne.

6. Blocs : procédure centrale

Tous les interrupteurs utiliseront la même procédure. Cela rend l’application plus propre et évite de répéter les mêmes tests de connexion.

```
Procédure SendCommand(cmd)
    si BluetoothClient1.IsConnected
        appeler BluetoothClient1.SendText(cmd)
    sinon
        appeler Notifier1.ShowAlert("Bluetooth non connecté")
```

À faire dans App Inventor

1. Ouvrir l’onglet Blocks.
2. Aller dans Procedures et créer une procédure nommée SendCommand.
3. Ajouter un paramètre nommé cmd.
4. Dans la procédure, créer un bloc if then else.
5. Mettre BluetoothClient1.IsConnected comme condition.
6. Dans then : envoyer le texte cmd. Dans else : afficher une alerte.

Blocs de l’heure

```
Quand ClockClock.Timer
    mettre LabelTime.Text à ClockClock.FormatDateTime(ClockClock.Now, "HH:mm:ss")
```

7. Blocs : connexion Bluetooth

Le bouton Appareils est un ListPicker. Avant de l’ouvrir, on remplit sa liste avec les appareils Bluetooth déjà appairés.

Avant l’ouverture du ListPicker

```
Quand ListPickerBluetooth.BeforePicking
    mettre ListPickerBluetooth.Elements à BluetoothClient1.AddressesAndNames
```

Après sélection d’un appareil

```
Quand ListPickerBluetooth.AfterPicking
    si BluetoothClient1.Connect(ListPickerBluetooth.Selection)
        mettre LabelStatus.Text à "Connected"
        mettre LabelStatus.TextColor à vert
        mettre LabelBluetoothDot.TextColor à vert
    sinon
        mettre LabelStatus.Text à "Not Connected"
        mettre LabelStatus.TextColor à rouge
        mettre LabelBluetoothDot.TextColor à rouge
```

Test minimal
À ce stade, l’élève doit déjà pouvoir ouvrir l’application, cliquer sur Appareils, choisir HC-05 / HC-06 et voir le statut passer en vert.

8. Blocs : interrupteurs

Chaque interrupteur envoie une lettre différente selon ON ou OFF. Les lettres doivent être identiques dans l'application et dans le code Arduino.

```
Quand SwitchSalon.Changed
    si SwitchSalon.On envoyer "s" sinon envoyer "S"

Quand SwitchCuisine.Changed
    si SwitchCuisine.On envoyer "c" sinon envoyer "C"

Quand SwitchWC.Changed
    si SwitchWC.On envoyer "h" sinon envoyer "H"

Quand SwitchChambre.Changed
    si SwitchChambre.On envoyer "w" sinon envoyer "W"
```

Toutes les lumières

```
Quand SwitchAll.Changed
    si SwitchAll.On
        appeler SendCommand("t")
        mettre SwitchSalon.On à true
        mettre SwitchCuisine.On à true
        mettre SwitchWC.On à true
        mettre SwitchChambre.On à true
    sinon
        appeler SendCommand("T")
        mettre SwitchSalon.On à false
        mettre SwitchCuisine.On à false
        mettre SwitchWC.On à false
        mettre SwitchChambre.On à false
```

Ventilateur

```
Quand SwitchFan.Changed
    si SwitchFan.On
        appeler SendCommand("v")
        mettre LabelFanStatus.Text à "En marche"
    sinon
        appeler SendCommand("V")
        mettre LabelFanStatus.Text à "Arrêté"
```

9. Réception température / luminosité

Arduino doit envoyer régulièrement une ligne simple. Le format recommandé est volontairement facile à découper dans App Inventor.

```
Format envoyé par Arduino :
LUM=450;TEMP=23.5
```

Principe des blocs

1. Dans ClockReceive.Timer, vérifier que Bluetooth est connecté.
2. Vérifier que BytesAvailableToReceive > 0.
3. Lire une ligne avec ReceiveText(-1).
4. Séparer le texte avec ; pour obtenir deux morceaux.
5. Séparer chaque morceau avec =.
6. Mettre à jour LabelLumValue et LabelTempValue.

```
Quand ClockReceive.Timer
    si BluetoothClient1.IsConnected ET BluetoothClient1.BytesAvailableToReceive > 0
        receivedText = BluetoothClient1.ReceiveText(-1)
        parts = split receivedText at ";"
        lumPart = item 1 of parts      // LUM=450
        tempPart = item 2 of parts     // TEMP=23.5
        lumValue = item 2 of split lumPart at "="
        tempValue = item 2 of split tempPart at "="
        LabelLumValue.Text = lumValue + " lux"
        LabelTempValue.Text = tempValue + " °C"
```

Sécurité
Si les valeurs ne s’affichent pas, afficher temporairement receivedText dans un label de debug pour voir exactement ce que le téléphone reçoit.

10. Code Arduino de référence

Ce code est un squelette. Les numéros de pins peuvent être adaptés au montage réel. L'important est que les lettres correspondent exactement à l'application.

```
#define LED_SALON   2
#define LED_CUISINE 3
#define LED_WC      4
#define LED_CHAMBRE 5
#define FAN_PIN     6

unsigned long lastSend = 0;

void setup() {
  Serial.begin(9600);          // HC-05 / HC-06 souvent à 9600
  pinMode(LED_SALON, OUTPUT);
  pinMode(LED_CUISINE, OUTPUT);
  pinMode(LED_WC, OUTPUT);
  pinMode(LED_CHAMBRE, OUTPUT);
  pinMode(FAN_PIN, OUTPUT);
}

void loop() {
  if (Serial.available()) {
    char cmd = Serial.read();

    if (cmd == 's') digitalWrite(LED_SALON, HIGH);
    if (cmd == 'S') digitalWrite(LED_SALON, LOW);

    if (cmd == 'c') digitalWrite(LED_CUISINE, HIGH);
    if (cmd == 'C') digitalWrite(LED_CUISINE, LOW);

    if (cmd == 'h') digitalWrite(LED_WC, HIGH);
    if (cmd == 'H') digitalWrite(LED_WC, LOW);

    if (cmd == 'w') digitalWrite(LED_CHAMBRE, HIGH);
    if (cmd == 'W') digitalWrite(LED_CHAMBRE, LOW);

    if (cmd == 't') {
      digitalWrite(LED_SALON, HIGH);
      digitalWrite(LED_CUISINE, HIGH);
      digitalWrite(LED_WC, HIGH);
      digitalWrite(LED_CHAMBRE, HIGH);
    }
    if (cmd == 'T') {
      digitalWrite(LED_SALON, LOW);
      digitalWrite(LED_CUISINE, LOW);
      digitalWrite(LED_WC, LOW);
      digitalWrite(LED_CHAMBRE, LOW);
    }

    if (cmd == 'v') digitalWrite(FAN_PIN, HIGH);
    if (cmd == 'V') digitalWrite(FAN_PIN, LOW);
  }

  if (millis() - lastSend > 1000) {
    lastSend = millis();
    int lumValue = analogRead(A0);
    float tempValue = 23.5;  // remplacer par lecture DHT si disponible
    Serial.print("LUM=");
    Serial.print(lumValue);
    Serial.print(";TEMP=");
    Serial.println(tempValue);
  }
}
```

11. Protocole de test

Ne jamais tester toute l'application en une seule fois. On valide une fonction, puis on ajoute la suivante.

Étape	Action	Résultat attendu
1	Installer AI Companion ou compiler l'APK	L'app démarre sur le téléphone
2	Appairer HC-05 / HC-06 dans Android	Le module apparaît dans les appareils Bluetooth
3	Cliquer Appareils dans l'app	La liste Bluetooth s'ouvre
4	Choisir HC-05 / HC-06	Le statut devient Connected en vert
5	Tester SwitchSalon	La sortie Salon change
6	Tester les trois autres zones	Chaque LED répond à sa lettre
7	Tester Toutes les lumières	Les quatre sorties changent ensemble
8	Tester Ventilateur	La sortie FAN change et le label change
9	Attendre les capteurs	Température et luminosité s'affichent

Erreurs fréquentes

Symptôme	Cause probable	Correction
Aucun appareil affiché	Bluetooth non appairé ou permission refusée	Appairer dans Android et autoriser Bluetooth
Connexion impossible	Mauvais module sélectionné ou baud rate incorrect	Tester HC-05/HC-06, vérifier 9600 bauds
LED ne répond pas	Lettre différente entre app et Arduino	Comparer la table des commandes
Capteurs vides	Arduino n'envoie pas avec println	Envoyer LUM=...;TEMP=... avec Serial.println
Caractères étranges	Mauvaise vitesse série	Vérifier Serial.begin et réglage du module

12. Grille de validation finale

Cette grille peut être utilisée par les élèves avant rendu ou par l’enseignant pour vérifier le projet.

Critère	Validation
Interface fidèle au module : titre, Bluetooth, capteurs, éclairage, ventilateur, aide	☐
Icône de l’application ajoutée	☐
Connexion Bluetooth fonctionnelle	☐
Statut visuel rouge / vert fonctionnel	☐
Salon, Cuisine, WC, Chambre répondent séparément	☐
Toutes les lumières fonctionne	☐
Ventilateur ON/OFF fonctionne	☐
Température s’affiche	☐
Luminosité s’affiche	☐
Aide intégrée avec les commandes	☐
APK ou projet App Inventor exporté	☐

Résultat final
L’application est validée lorsque le téléphone contrôle réellement les sorties Arduino et reçoit les valeurs de capteurs en direct.

Rappel des composants non visibles	BluetoothClient1, ClockClock, ClockReceive, Notifier1
URL tutoriel	https://help.futurisn.com/1.html