MIT App Inventor Codi Bot: Wing control

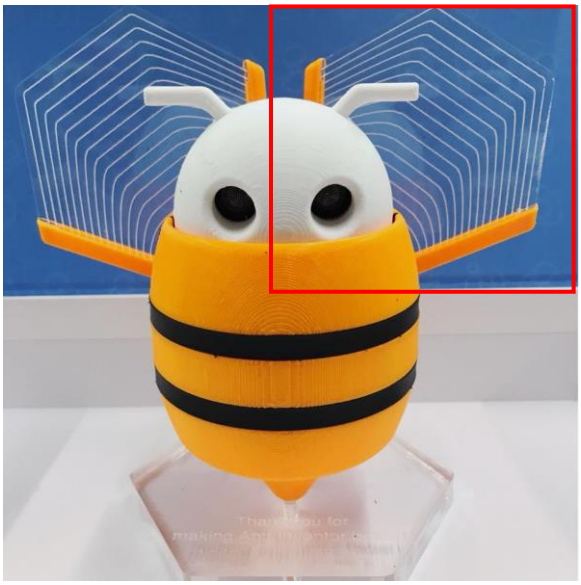
60 mins

Level: advanced

This tutorial will help you get started with App Inventor + IoT, control one wing of Codi Bot by buttons and slider. We also provide a complete app for you to control both Codi Bot wings.

Note that there may be slight difference due to the tolerance of each servo, you may have to modify the parameters for servo position. Please first check everything is assembled correctly according to the **Codi Bot Standalone Demo tutorial**.

- **source .ino** / **source .aia**
- **complete .aia**



Function description

This project will show you how to control one of Codi Bot wings, which is actually a small servo motor, with App Inventor through BLE communication. The components used in this tutorial are buttons and a slider.

Hardware

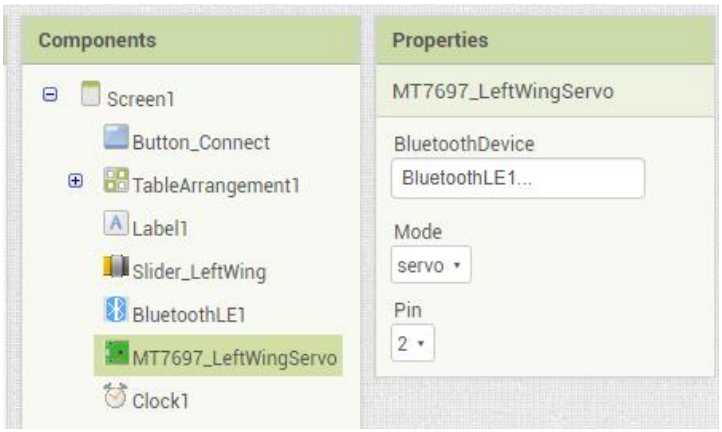
Please follow this **building guide** to assemble your Codi Bot.

App Inventor

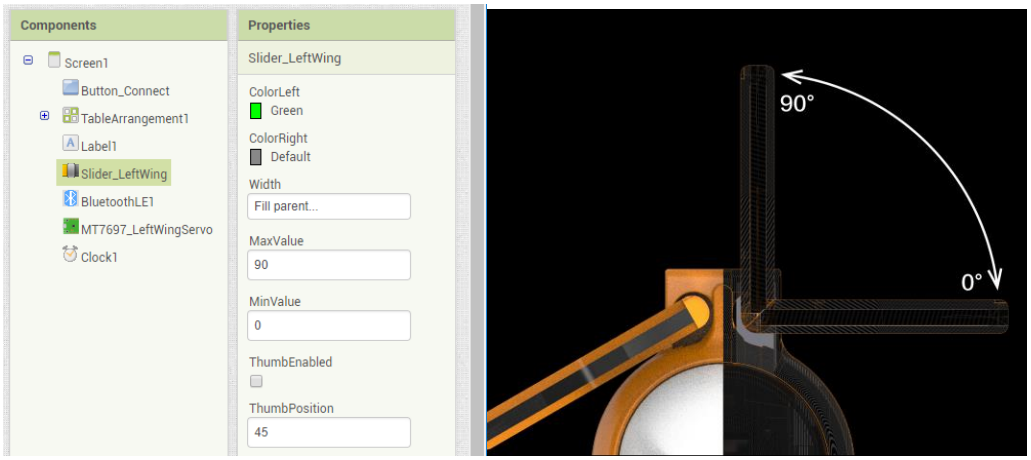
Now log in to your App Inventor account and create a new project or directly import **this aia file**.

Designer

1. We need to import two extensions from this URL:
    - **Bluetooth low energy**: http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.ble.aix
    - **MT7697pin**: http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.iot.mt7697.aix
2. Add a **BluetoothLE** component to your project, we use it to send commands to Codi Bot through Bluetooth communication.
3. Add a **MT7697pin** component to your project, we use them to control a pin of LinkIt 7697, where is also the servo motor connected to.
    - Rename one **MT7697pin** component as "**MT7697_LeftWingServo**". Set the BluetoothDevice property to **BluetoothLE1** (Step 2.), Mode to **servo** and set Pin to **2**. This is because we have connected the servo signal pin to LinkIt 7697 #2 pin.

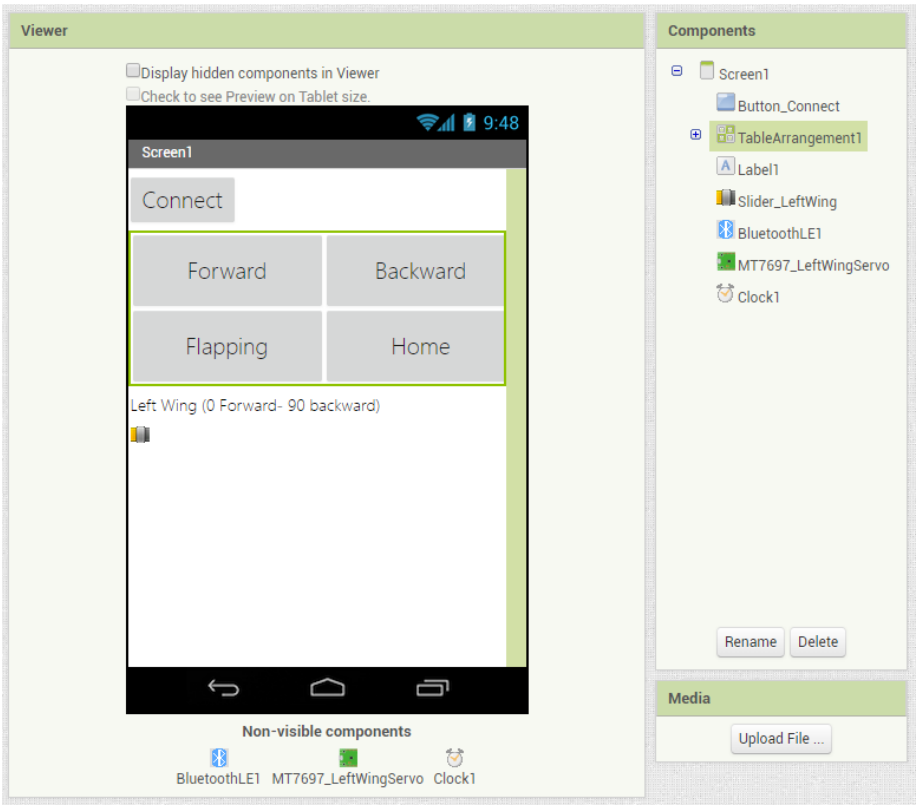


4. Add a button to establish Bluetooth connection between your Android phone and LinkIt 7697. Rename it as "**Button_Connect**" and set Text to "**Connect**".
5. Add another four buttons for corresponding wing action. Rename them as "**Button_WingForward**", "**Button_WingBackward**", "**Button_wingFlap**" and "**Button_Home**". Set FontSize to **20**, Width to 50 percent and Text to "**Forward**", "**Backward**", "**Flaping**" and "**Home**" accordingly.
6. Add a TableArrangement component, set both Columns and Rows to **2**. Then put buttons of Step 5 into it.
7. Add a label to show message. Set its Text to "**Left Wing (0 Forward- 90 backward)** ".
8. Add a slider to control the servo motor position. Rename it as "**Slider_LeftWing**". Set its MaxValue to **90**, MinValue to **0** and ThumbPosition to **45**. this is because after the servo is installed, it can only move in the range of 0 - 90 degree.



9. Add a Clock component, uncheck the TimerEnabled property and set TimerInterval to **1000** (1 second). We use this time duration to make sure the wing can move to its destination.

After some adjusting, your designer should look similar to the image below. It doesn't have to be exactly the same. Feel free to modify the component's background color, position and text size.
Note: The TableArrangement and components insides are shown here.



Blocks

Let's take a look at our blocks step by step:

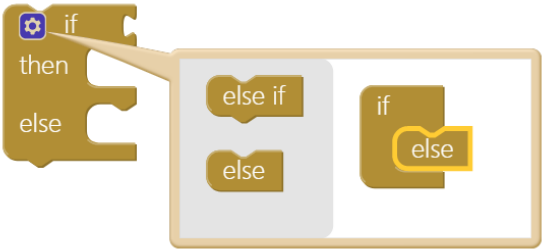
1. **Variable for Bluetooth address**

Please replace the **addr** variable value with what you get from Arduino Serial Monitor, this is the Bluetooth address of LinkIt 7697. We will show you how to check this information in [Arduino IDE and sketch] section.

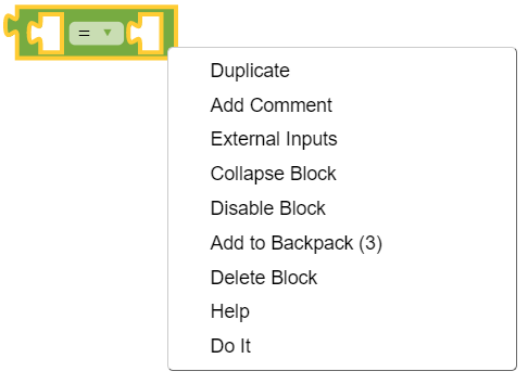
2. **Initialize app and scan for nearby Bluetooth devices**

In **Screen1.Initialize** event, we ask **BluetoothLE** component to scan for BLE devices nearby (**BluetoothLE1.StartScanning**).

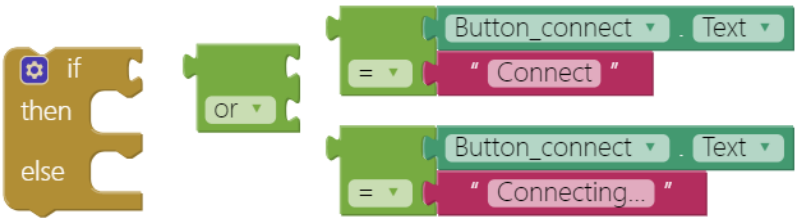
3. **Connect and disconnect from Bluetooth device**
   In **Button_Connect.Click** event, we are going to connect or disconnect from Bluetooth device depending on the button text. First, add an **if** condition, then click its blue gear icon to add an **else**.



   Add an **or** command from logic block, then right-click it and select "**External Inputs**". This will make it into more rectangular shape with input on the right hand side.

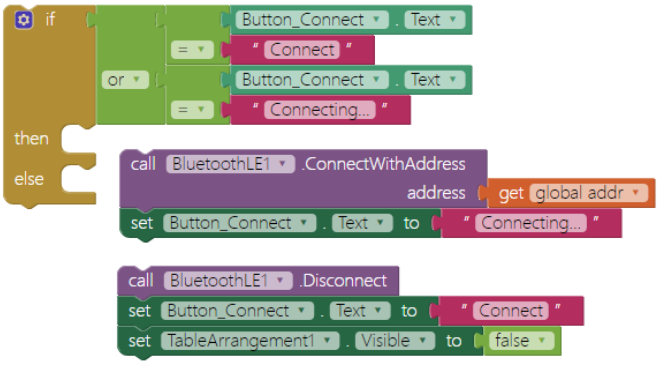


   Now we want to check whether the **Button_Connect.Text** status is "**Connect**" OR "**Connecting...**", this is how App Inventor decide to connect or disconnect Bluetooth connection with LinkIt 7697. please combine these blocks.

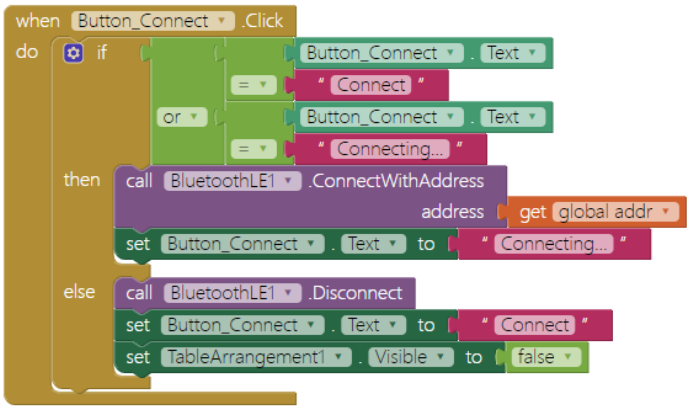


Good! When the **Button_Connect**'s text reads "**Connect**" or "**Connecting...**", the app will connect to the specified Bluetooth device (**BluetoothLE1.ConnectwithAddress**), which is our LinkIt 7697.

If the text does not read "**Connect**", first disconnect (**BluetoothLE1.Disconnect**) and show message on Button_Connect, then set slider Enabled property to **false**.



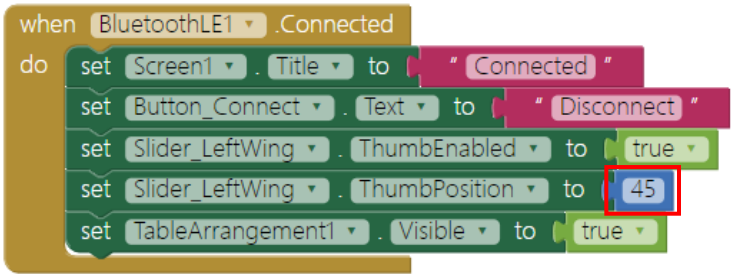
Put everything into **Button_Connect.Click** event, and finish like this:



4. **BLE Connected**

   When connected successfully (**BluetoothLE.Connected** event),

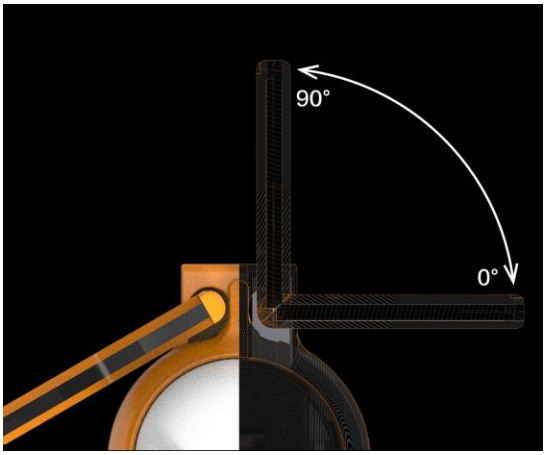
we will see related messages on several components. Note that we set **Slider_LeftWingServo.ThumbPosition** to 45, this will cause the left wing to move to the position of 45 degree, which is its home position.

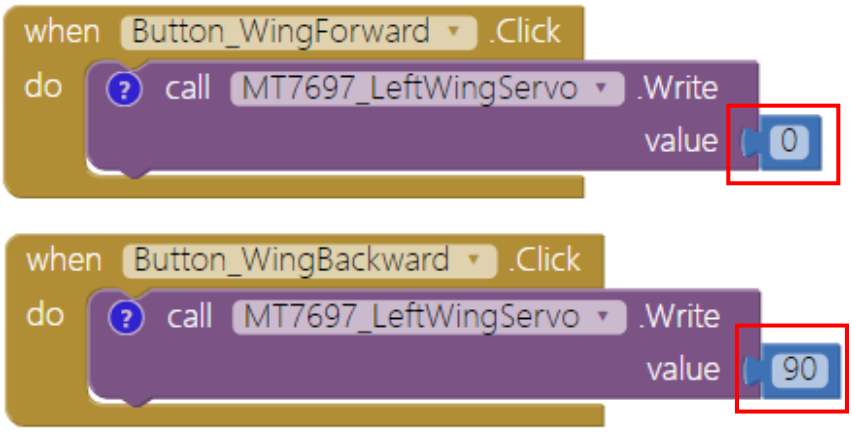


5. **Buttons to make wing moving forward and backward**

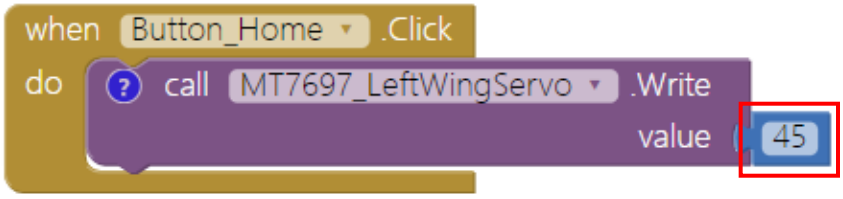
   We use several buttons to make the servo motor move to the position we want.

   When **Button_WingForward** is pressed, we make the servo to move to position **0** by **MT7697_LeftWingServo.Write** method. Similarly, when **Button_WingForward** is pressed, we make the servo to move to position **90**.



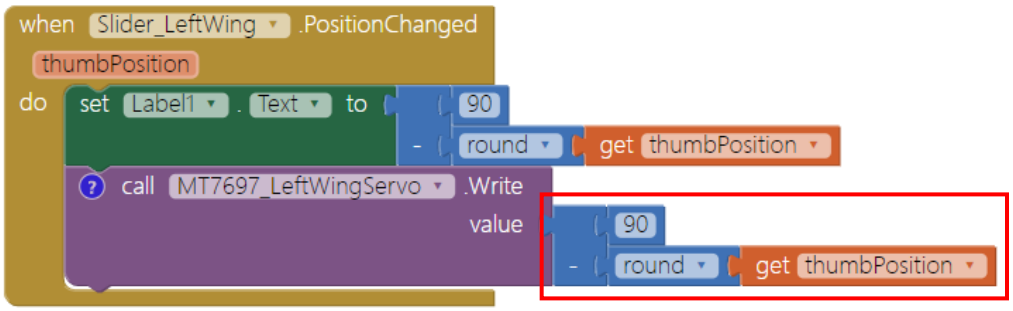


6. **Button_Home to make wing to its home position**
   Similar to previous step, we move the left wing servo to its home position (45 degree) when **Button_Home** is pressed. This is the same action when it is connected (Step 4.).



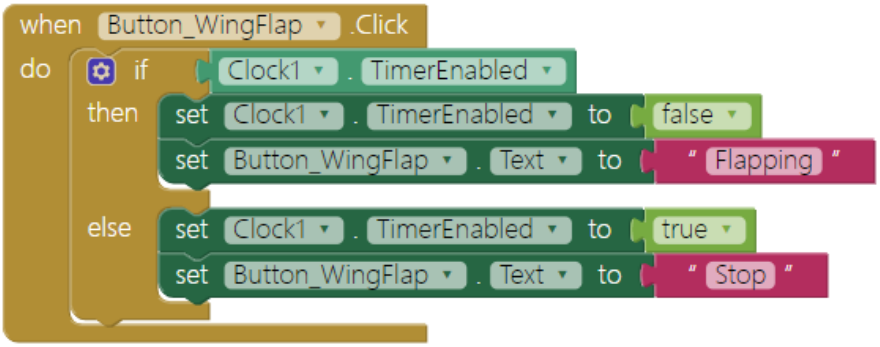
7. **Drag slider to move the wing**
   When the slider is dragged (**Slider_LED_R.PositionChanged** event), we show the position on label and tell the left wing to move to this position: (**90 minus thumbPosition**), this is because the 90 degree position of slider and wing are in the opposite direction.

   Note that we've use a math **round** method to this is because this method can apply integer only.



8. **Button to turn on/off the clock timer**

We use **Button_WingFlap** to turn on/off the clock timer, which consequently make the wing flapping/stop. In **Button_WingFlap** event, we first check whether the Timer is on: if it is on, then turn off the timer and change button text to "**Flapping**"; if it is off, then turn on the timer and change text to "**Stop**".

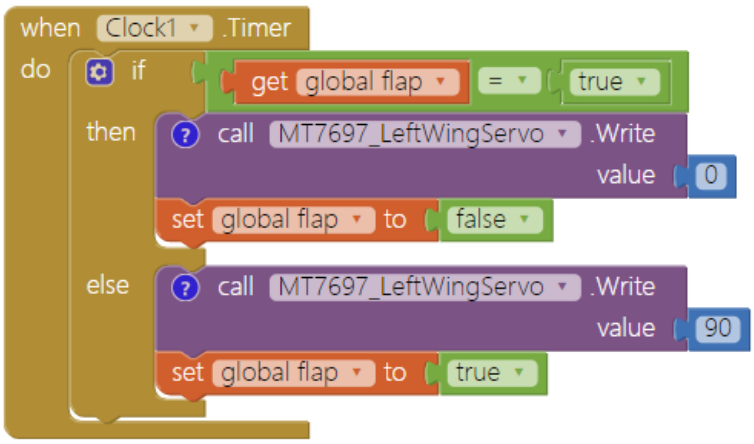


9. **wing flapping/stop**

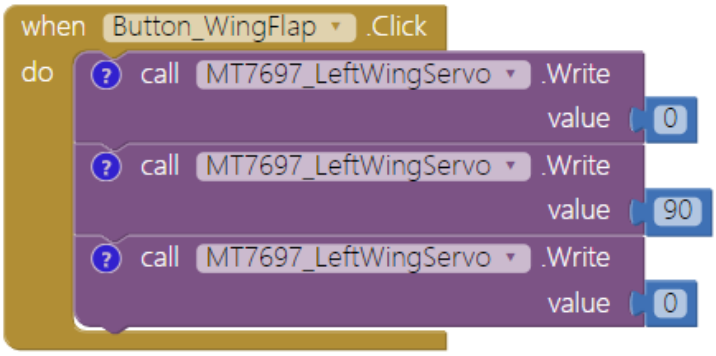
This is the most interesting function of this project. We use a logic variable named **flap** to tell servo whether it is allowed to move.

In **Clock.Timer** event, we check whether **flap** variable is true. If it is **true**, then make the servo move to position 0 (outward) and set the variable to **false**; if it is **false**, then move to position **90** (backward) and set the variable to **true**.

Excellent, Codi Bot is flapping its left wing!

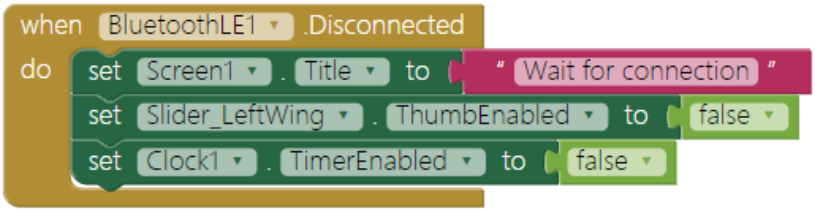


Note: if you do thing like below, the servo will do these three thing in a instant, which seems like the servo has gone to position 0 directly and never been to position 90.



10. Disconnect

After Bluetooth communication is closed successfully (trigger in Step1), we reset the app to its initial state to wait for next connect request in **BluetoothLE1.disconnected** event.



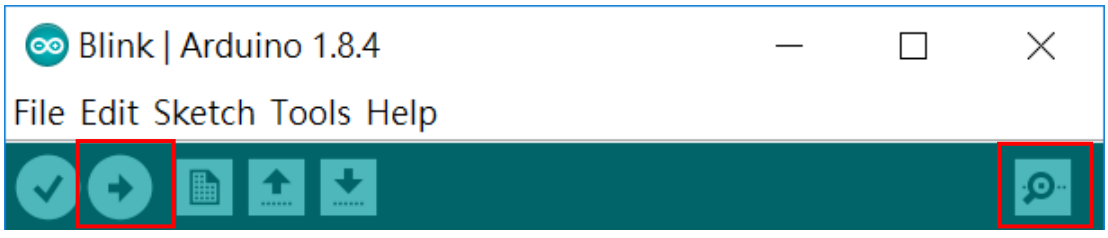
Arduino IDE and sketch

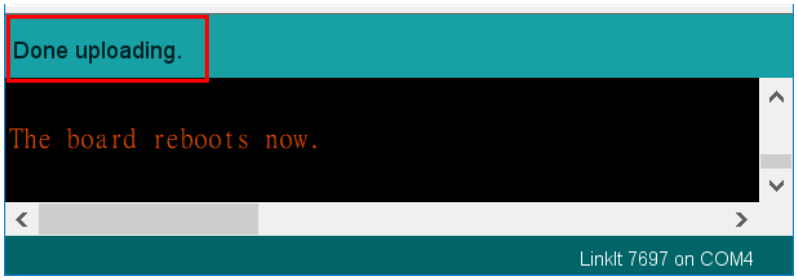
Make sure your computer has Arduino IDE installed and LinkIt 7697 SDK/driver are ready. If not, please check Codi Bot Standalone tutorial.

Connect your computer and LinkIt 7697 with a microUSB cable.

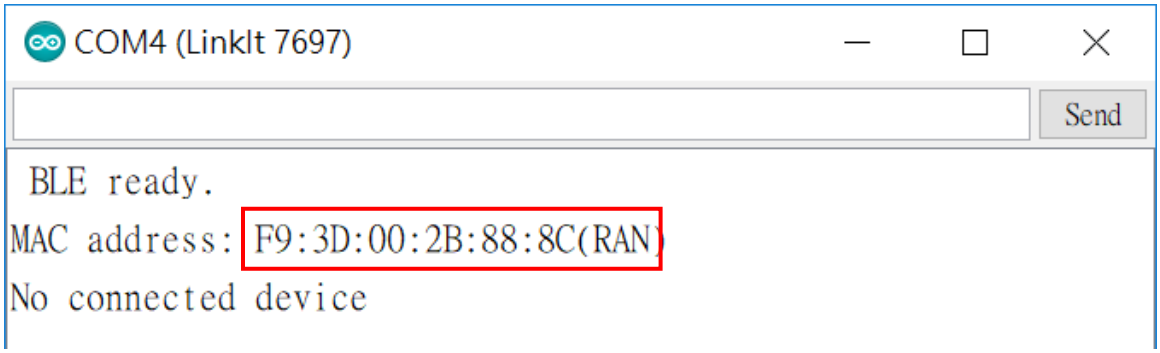
Please download the Arduino sketch from **here** and open in your Arduino IDE. This sketch can be used for all following Codi Bot projects, let you focus on building App Inventor projects.

Press the "**Upload**" right-arrow button of Arduino IDE, this will compile and upload the Arduino sketch to your LinkIt 7697. Please make sure you see the "**done uploading**" message in the console.



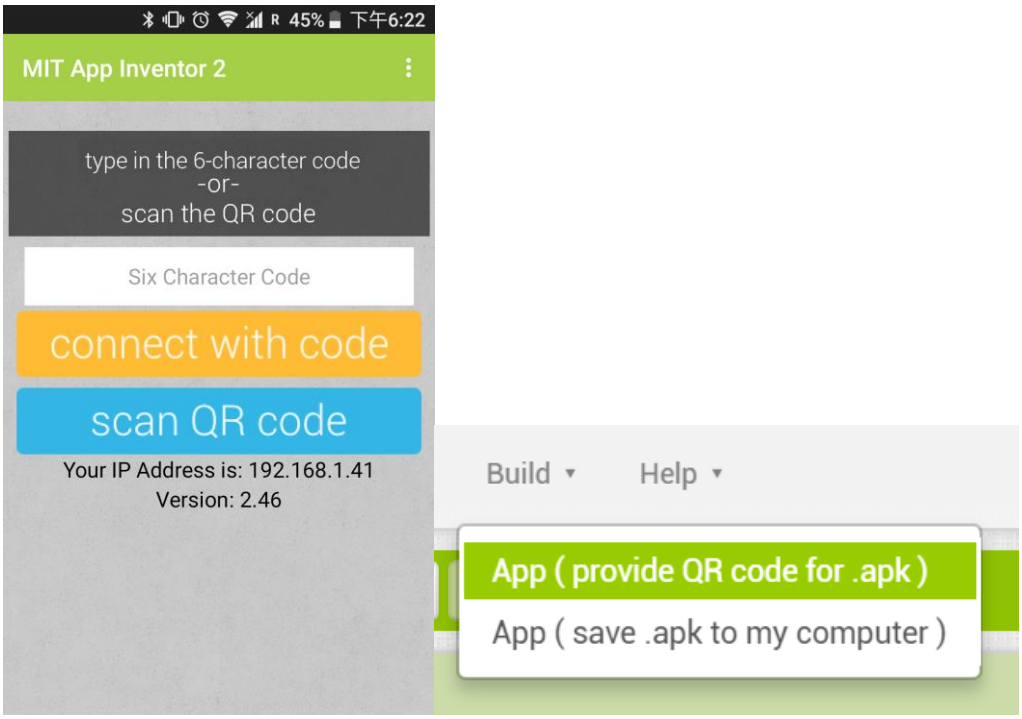


Click the magnifier icon at the up-right corner of Arduino IDE, you should see a message in the pop-up window. The [**XX:XX:XX:XX:XX:XX**] 12-digit string is the Bluetooth address of your LinkIt 7697. We have to modify the **addr** variable value of your AI2 project.



Tips

Make sure your LinkIt 7697 is running correctly. And install App Inventor project on your Android phone by clicking Build / App (provide QR code for .apk), this will show a qrcode for .apk file. Use MIT AI2 Companion to scan this qrocde, download and install.



Open your app (Fig. 1) and click **Connect** button. You'll see a "**Connected**" message on screen title if your phone is connected with Codi Bot correctly.

Click **Forward** and **Backward** button, Codi Bot's left wing should move correspondingly. Also click **Flap** button, the wing will move back and forth about every one second (Fig.2).

Finally drag the slider, the wing will move to the position according to where you drag the slider. Don't drag to fast or you may damage the servo. It takes time for servo to move to a specific position.

Remember to click **Disconnect** button when you finish with this project.

Fig 1. Initial screen	Fig 2. Press "**Flapping**" button to flap the wing (see its text had

	changed to "**Stop**").

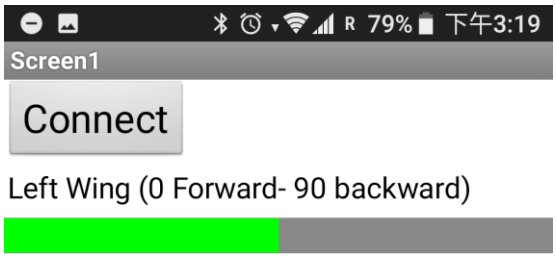
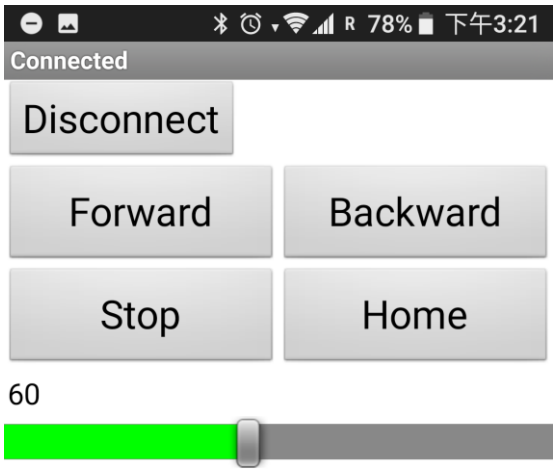
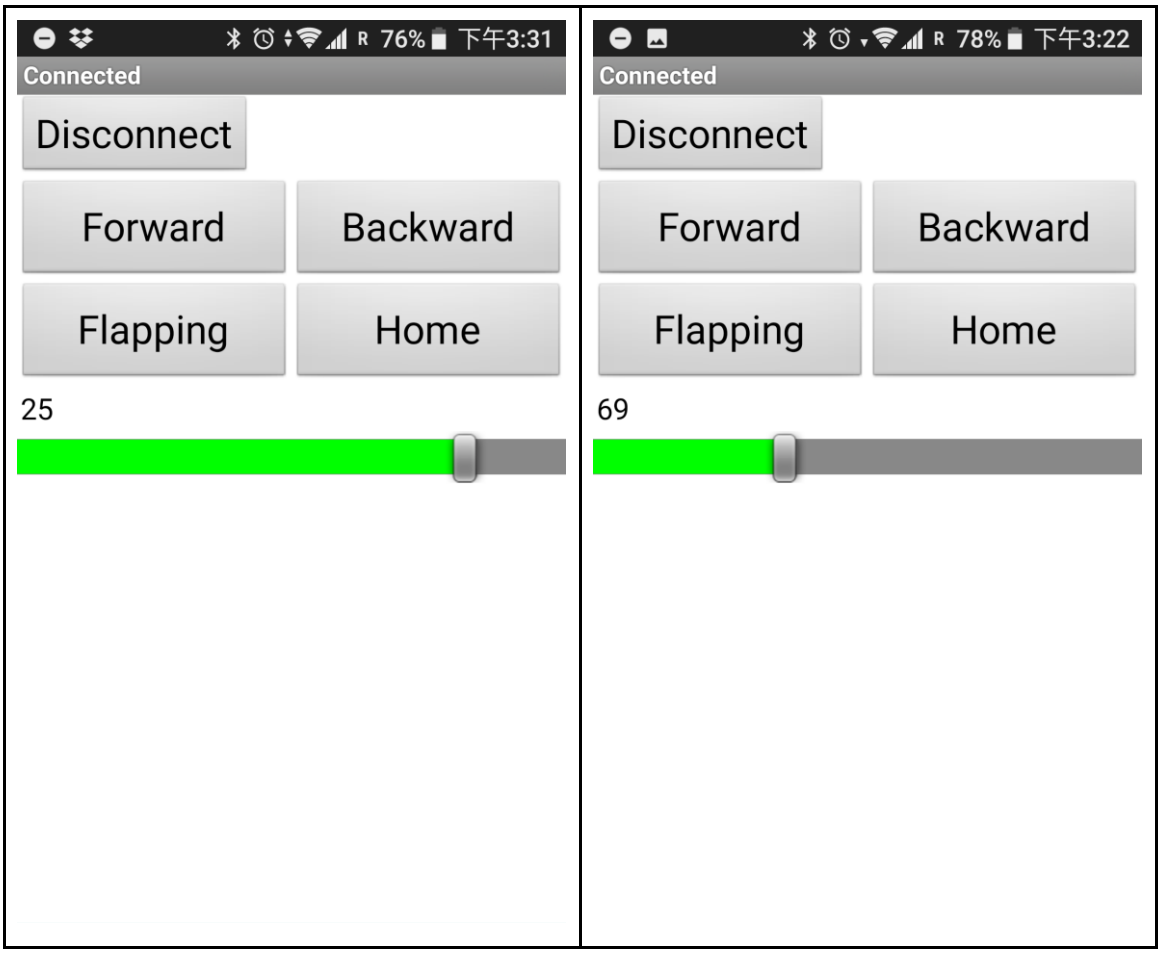
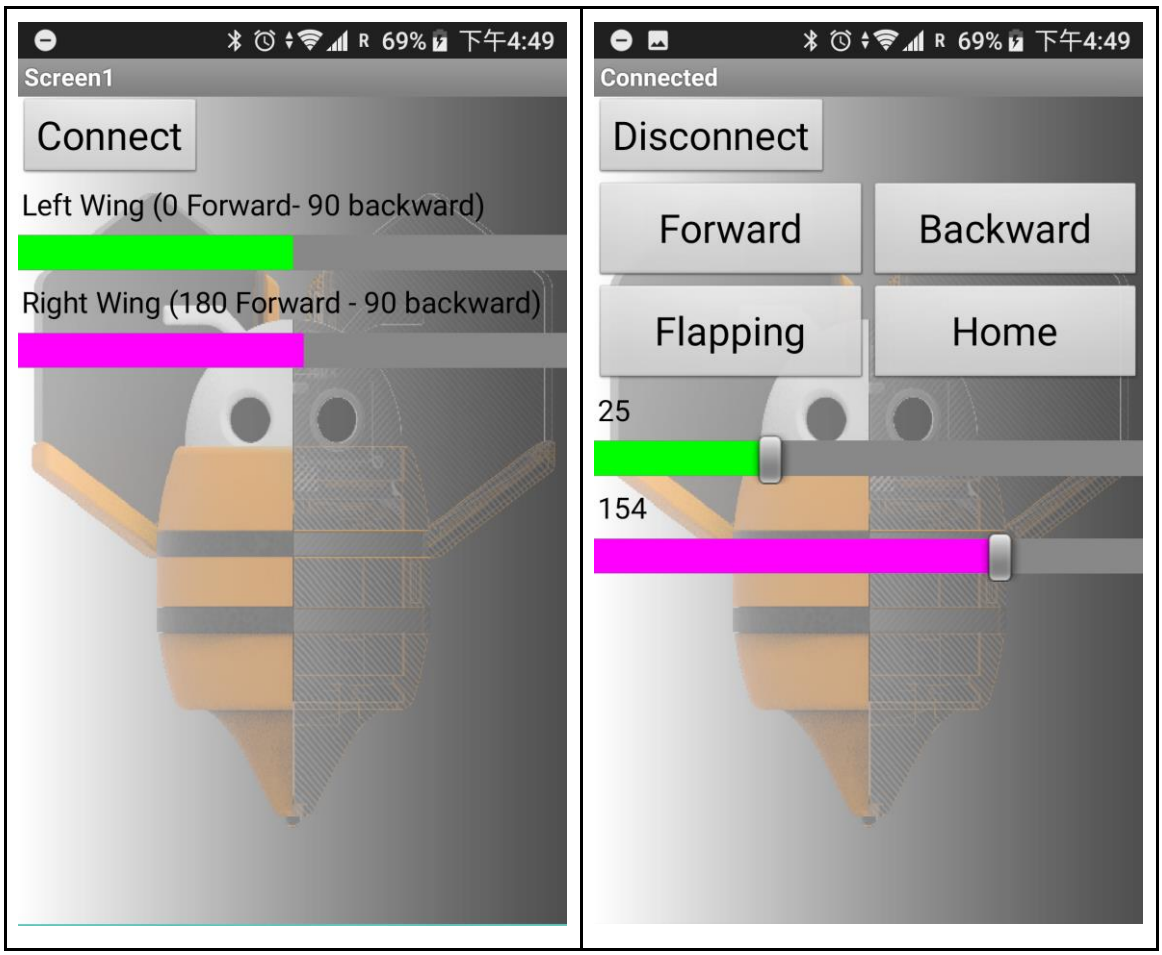

	

Fig 3. Drag slider to the right (**25** degree)	Fig 4. drag slider to the left (**69** degree)


76% 下午3:31
Connected
Disconnect
Forward
Backward
Flapping
Home
25
78% 下午3:22
Connected
Disconnect
Forward
Backward
Flapping
Home
69

Complete wing control app

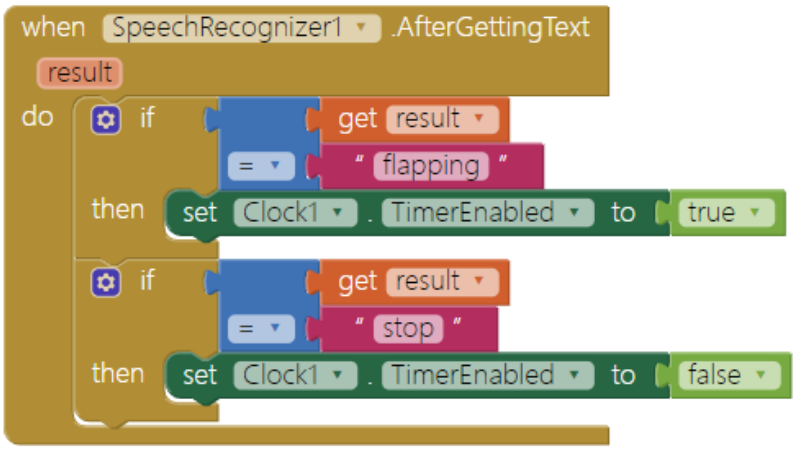
We have provided a complete app to control both Codi Bot wings, please import this **complete .aia** to your App Inventor.



Brainstorming

1. Use App Inventor **SpeechRecognizer** component to make wing flapping/stop.

*Hint: In **SpeechRecognizer.AfterGettingText** event, if the recognized result is "flapping" then turn on the Clock timer, or if it is "stop" then turn off. Other things in **Clock.Timer** event are unchanged.*


when SpeechRecognizer1 .AfterGettingText
result
do
if
get result
=
" flapping "
then
set Clock1 . TimerEnabled to true
if
get result
=
" stop "
then
set Clock1 . TimerEnabled to false