MIT App Inventor Codi Bot: Complete

90
mins

Obsolete
May Not Work

Level: advanced

This tutorial combines all the previous tutorials together. Here, you can interact with all Codi Bot functions, including LEDs, wings and sounds.

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

Obsolete
May Not Work

Function description

This project will show you how to control all Codi Bot functions with App Inventor through a BLE connection.

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**.

Since this app is the combination of the previous three, this tutorial will not go into very much detail. But you are certainly can follow this tutorial to build the app step by step.

This app is composed of **four** different sections:

1. Connection (**Designer**/**Blocks**)
2. Wing (**Designer**/**Blocks**)
3. LED (**Designer**/**Blocks**)
4. Buzzer (**Designer**/**Blocks**)

Designer

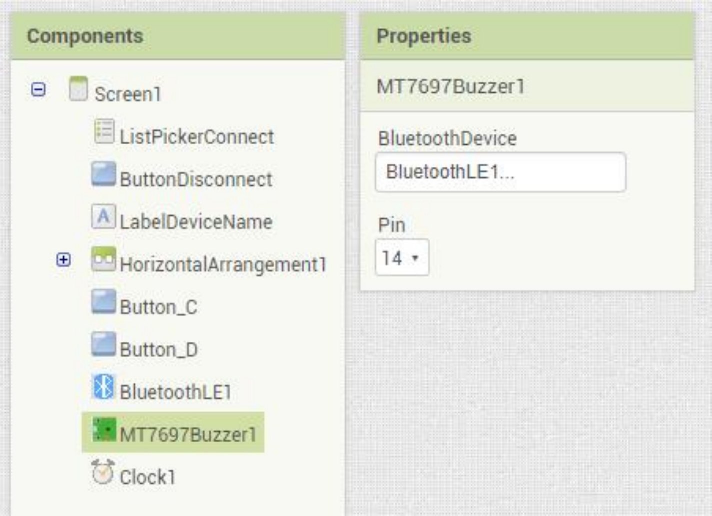
1. We need to import two extensions from this URL:
    - **Bluetooth low energy**: http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.ble.aix
    - **MT7697pin**: http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.iot.mt7697.aix
2. Add a **BluetoothLE** component to your project, we will use it to send commands to Codi Bot through Bluetooth connection.

Obsolete May Not Work

3. Add seven **MT7697Pin** components to your project. Set their BluetoothDevice properties to **BluetoothLE1** and rename them as table below:

Rename as	**Mode**	**Pin**
MT7697_LeftWingServo	servo	2
MT7697_RightWingServo	servo	7
MT7697_LeftWingLED	analog output	3
MT7697_RightWingLED	analog output	9
MT7697_LED_R	analog output	15
MT7697_LED_G	analog output	16
MT7697_LED_B	analog output	11

4. Add an **MT7697Buzzer** component to your project, we use this to control the #14 pin of LinkIt 7697, which is connected with Robot Shield onboard buzzer.
    - Set its **BluetoothDevice** property to **BluetoothLE1** (Step 2.) and set **Pin** to **14**.



Components

- Screen1
    - ListPickerConnect
    - ButtonDisconnect
    - LabelDeviceName
    - HorizontalArrangement1
    - Button_C
    - Button_D
    - BluetoothLE1
    - MT7697Buzzer1
    - Clock1

Properties

MT7697Buzzer1

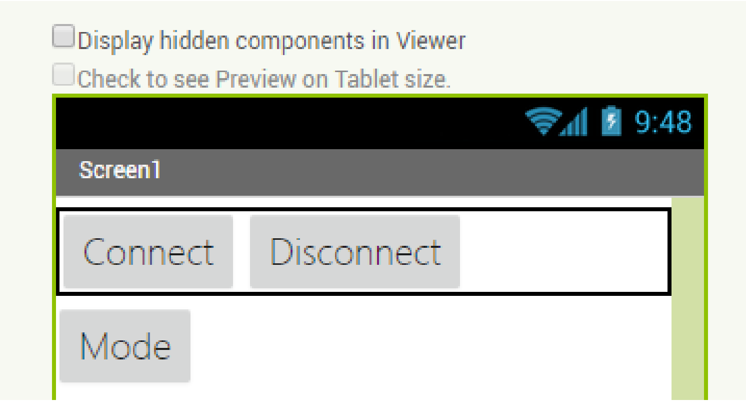
BluetoothDevice: BluetoothLE1...

Pin: 14

Obsolete May Not Work

Connection

5. Add a ListPicker to select available Bluetooth devices nearby. Rename it to "**ListPicker_Connect**", set **Text** to "**Connect**" and **FontSize** to **20**.
6. Add a button to close Bluetooth connection between your Android phone and LinkIt 7697. Rename it to "**Button_Disconnect**" and set **Text** to "**Disconnect**".
7. Add a Label to show messages of connection. Rename it to "**Label_Device**" and clear **Text** to "".
8. Add a HorizontalArrangement component, set Width to **Fill parent**, then add components from Step 4 to 6 into it.
9. Add another ListPicker to select one of the three control mode: Wing, LED and Sound. Rename it to "**ListPicker_Connect**", **Text** to "**Mode**" and **FontSize** to **20**.



☐ Display hidden components in Viewer

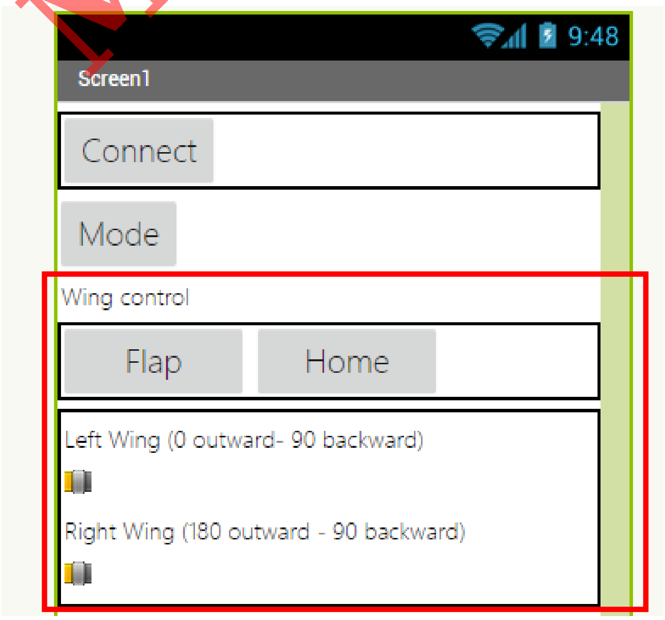
☐ Check to see Preview on Tablet size.

Wing (Designer)

10. Add a Label to show message, rename it to "**Label_Wing**" and set **Text** to "**Wing control**".
11. Add two Buttons for wing actions. Rename them to "**Button_WingFlap**" and "**Button_Home**". Set **FontSize** to **20** and **Text** to "**Flap**" and "**Home**".
12. Add a HorizontalArrangement component, set **Width** to **Fill parent**, then put components of Step 11 into it.

Obsolete May Not Work

13. Add two labels to show messages. Rename them to "**Label_LeftWing**" and "**Label_RightWing**". Set their **Text** to "**Left Wing (0 outward- 90 backward)** " and "**Right Wing (180 outward - 90 backward)** ".
14. Add a slider to control left wing position. Rename it to "**Label_LeftWing**". Set **MinValue** to **0**, **MaxValue** to **90**, **Width** to **Fill parent** and **ColorLeft** to Green.
15. Add a slider to control right wing position. Rename it to "**Label_RightWing**". Set **MinValue** to **90**, **MaxValue** to **180**, **Width** to **Fill parent** and **ColorLeft** to **Purple**.
16. Add a VerticalArrangement component, set **Width** to **Fill parent**, then add components from Step 13 to 15 into it.
17. Wing section finishes as below:


9:48
Screen1
Connect
Mode
Wing control
Flap
Home
Left Wing (0 outward- 90 backward)
Right Wing (180 outward - 90 backward)

LED (Designer)

18. Add a Label to show message, rename it to "**Label_LED**" and set **Text** to "**RGB LED & Wing LED**".
19. Add four buttons for RGB LED actions. Rename them to "**Button_LED_R**", "**Button_LED_G**", "**Button_LED_B**" and "**Button_RGB_Random**". Set **Text** to "**R**", "**G**", "**B**", and "**Random**"; **Width** to **30 percent**; and **FontSize** to **20**.

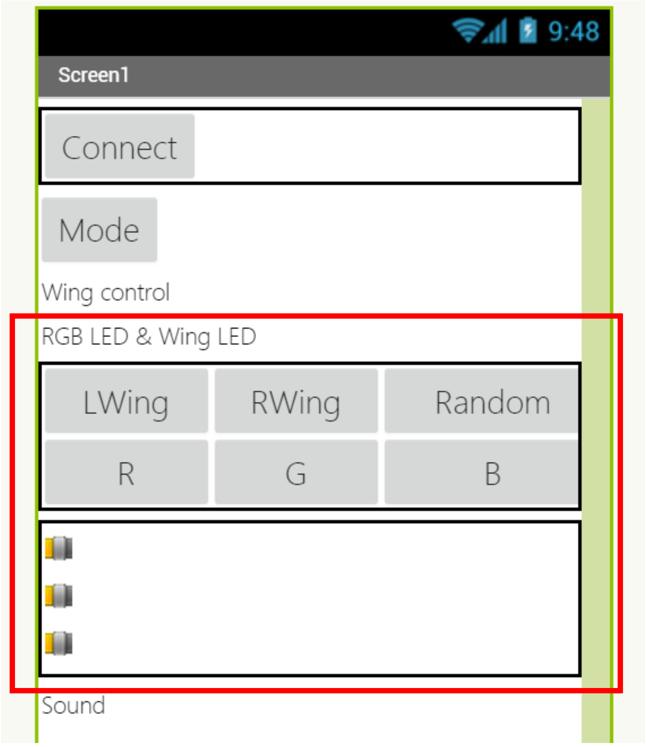
20. Add two more buttons for wing LEDs actions. Rename them as "**Button_LED_LeftWing**" and "**Button_LED_RightWing**". Set Text to "**L**" and "**R**", Width to **30 percent** and FontSize to **20**.

21. Add a TableArrangement component, set **Width** to **Fill parent**, **Columns** to **3** and **Rows** to **2**. Then add the components from Step 19 to 20 into it.

22. Add three sliders to control red/green/blue light intensity of RGB LED. Rename them to "**Slider_LED_R**", "**Slider_LED_G**" and "**Slider_LED_B**". Set ColorLeft to **Red**, **Green** and **Blue**; Set **MinValue** to **0**, **MaxValue** to **255** and **Width** to **Fill parent.**

23. Add a VerticalArrangement component, set **Width** to **Fill parent**, then add components of Step 22 into it.

24. LED section finishes as below:

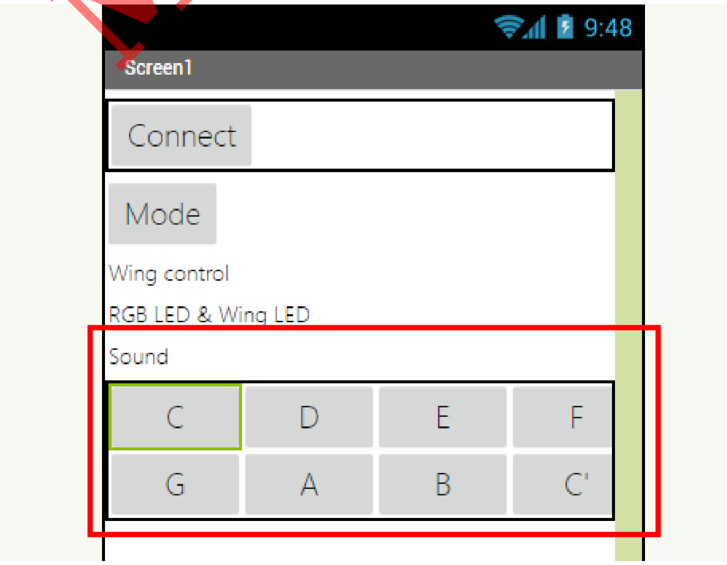


Obsolete
May Not Work

Obsolete May Not Work

Sound (Designer)

25. Add a Label to show message, rename it to "**Label_Sound**" and set Text to "**Sound**”.
26. Add eight buttons to make sounds. Rename them to "**Button_C**", "**Button_D**", "**Button_E**", "**Button_F**", "**Button_G**", "**Button_A**", "**Button_B**" and "**Button_C2**". **Text** to "**C**", "**D**", "**E**", "**F**", "**G**", "**A**", "**B**" and "**C’** ". Set **Width** to **25 percent** and **FontSize** to **20**.
27. Add a TableArrangement component, set **Width** to **Fill parent**, **Columns** to **4** and **Rows** to **2**. Then add the components of Step 26 into it.
28. Sound section finishes as below:



Not all the functions of previous tutorials are included here, for instance specifying sound frequency and time duration. However, you can always tinker with this app and add as many App Inventor components as you like.

Obsolete May Not Work

Blocks

Let’s take a look at our blocks step by step. Notice this time we have a different connection approach than with previous projects. We use ListPicker to select available BluetoothLE device instead of a specified device.

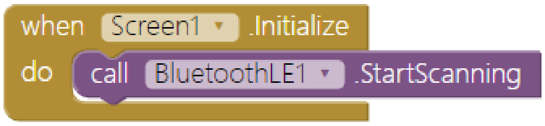
Similar to Designer, we use the same structure to guide you to finish the blocks needed for this project.

Connection (Blocks)

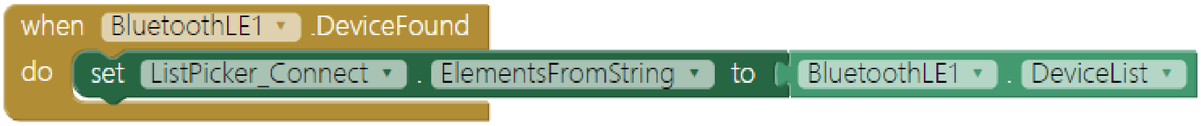
1. **Initialize app and scan for nearby Bluetooth devices**

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

   If any devices are found (**BluetoothLE1.DeviceFound** event), we put these devices into ListPicker.



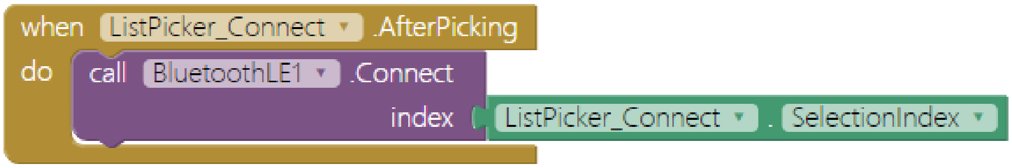
```
when Screen1 .Initialize
do  call BluetoothLE1 .StartScanning
```



```
when BluetoothLE1 .DeviceFound
do  set ListPicker_Connect . ElementsFromString to  BluetoothLE1 . DeviceList
```

2. **Choose device with ListPicker**

   We choose device to connect in a ListPicker. In the **ListPicker.AfterPicking** event, we use the BluetoothLE.Connect method to connect with the device selected.



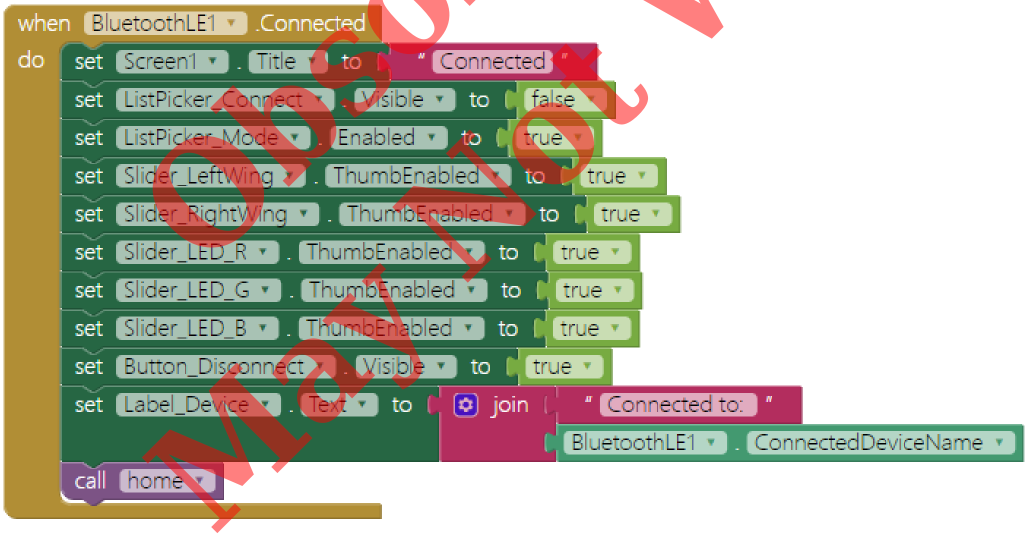
```
when ListPicker_Connect .AfterPicking
do  call BluetoothLE1 .Connect
                       index  ListPicker_Connect . SelectionIndex
```

3. **BLE Connected**

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

Obsolete May Not Work

we will see related messages on several components, hide the ListPicker and show other components. This is because if we send Bluetooth commands without connecting to something first, it may cause error.

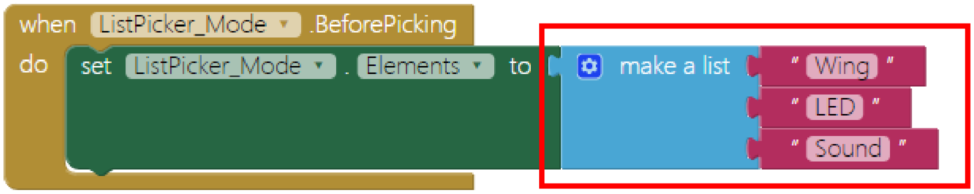


```
when BluetoothLE1 .Connected
do  set Screen1 . Title to " Connected "
    set ListPicker_Connect . Visible to false
    set ListPicker_Mode . Enabled to true
    set Slider_LeftWing . ThumbEnabled to true
    set Slider_RightWing . ThumbEnabled to true
    set Slider_LED_R . ThumbEnabled to true
    set Slider_LED_G . ThumbEnabled to true
    set Slider_LED_B . ThumbEnabled to true
    set Button_Disconnect . Visible to true
    set Label_Device . Text to join " Connected to: "
                                    BluetoothLE1 . ConnectedDeviceName
    call home
```

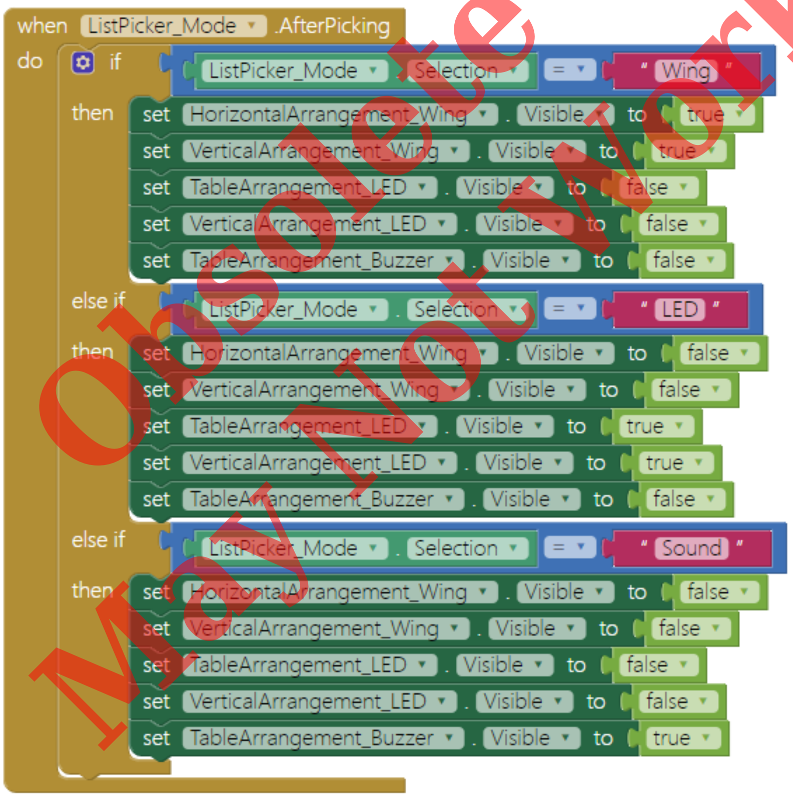
4. **Another ListPicker to choose one of the modes**

   To simplify the interface, we separate related components into three groups: Wing, LED and Sound. In the **ListPicker.BeforePicking** event, we set the ListPicker elements to a list (Wing, LED, Sound).

   Next, in **ListPicker.AfterPicking** event, we set related components to be visible or not according to the selection result.



```
when ListPicker_Mode .BeforePicking
do  set ListPicker_Mode . Elements to make a list " Wing "
                                                  " LED "
                                                  " Sound "
```



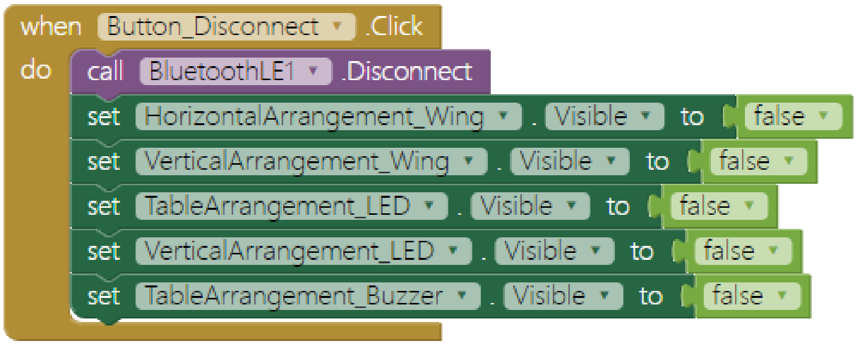
```
when ListPicker_Mode .AfterPicking
do  if   ListPicker_Mode . Selection = " Wing "
    then set HorizontalArrangement_Wing . Visible to true
         set VerticalArrangement_Wing . Visible to true
         set TableArrangement_LED . Visible to false
         set VerticalArrangement_LED . Visible to false
         set TableArrangement_Buzzer . Visible to false
    else if ListPicker_Mode . Selection = " LED "
    then set HorizontalArrangement_Wing . Visible to false
         set VerticalArrangement_Wing . Visible to false
         set TableArrangement_LED . Visible to true
         set VerticalArrangement_LED . Visible to true
         set TableArrangement_Buzzer . Visible to false
    else if ListPicker_Mode . Selection = " Sound "
    then set HorizontalArrangement_Wing . Visible to false
         set VerticalArrangement_Wing . Visible to false
         set TableArrangement_LED . Visible to false
         set VerticalArrangement_LED . Visible to false
         set TableArrangement_Buzzer . Visible to true
```

Obsolete May Not Work

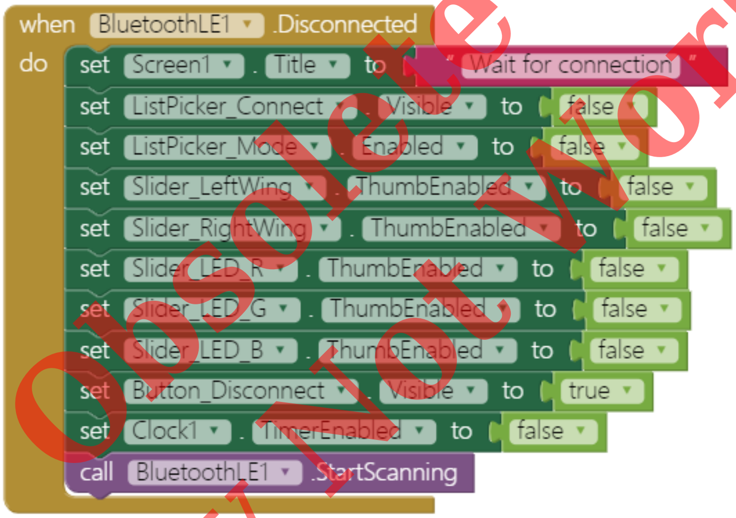
5. **Button to disconnect**

When **Button_Disconnect** is pressed, we close the Bluetooth connection and hide all Layout components.

And when we disconnect successfully (**BluetoothLE.Disconnected** event), we reset all components to their initial status and wait for the next connection.



```
when Button_Disconnect .Click
do  call BluetoothLE1 .Disconnect
    set HorizontalArrangement_Wing . Visible to false
    set VerticalArrangement_Wing . Visible to false
    set TableArrangement_LED . Visible to false
    set VerticalArrangement_LED . Visible to false
    set TableArrangement_Buzzer . Visible to false
```



```
when BluetoothLE1 .Disconnected
do  set Screen1 . Title to " Wait for connection "
    set ListPicker_Connect . Visible to false
    set ListPicker_Mode . Enabled to false
    set Slider_LeftWing . ThumbEnabled to false
    set Slider_RightWing . ThumbEnabled to false
    set Slider_LED_R . ThumbEnabled to false
    set Slider_LED_G . ThumbEnabled to false
    set Slider_LED_B . ThumbEnabled to false
    set Button_Disconnect . Visible to true
    set Clock1 . TimerEnabled to false
    call BluetoothLE1 .StartScanning
```

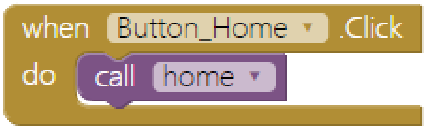
Obsolete
May Not Work

Wings (Blocks)

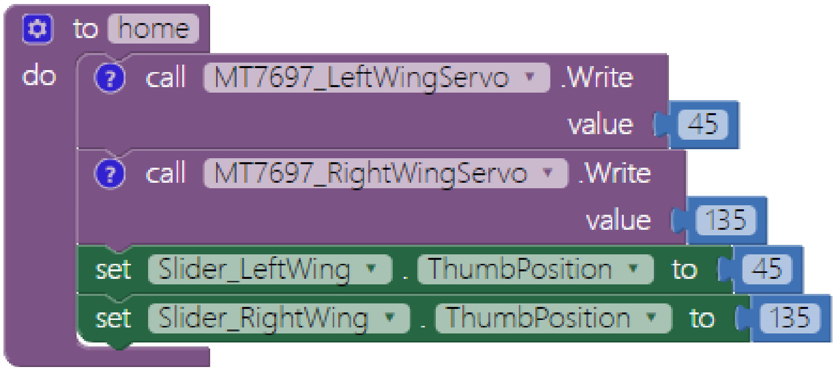
Here we will control the wings by buttons and sliders, as we did in the **Codi Bot wing tutorial**.

6. Button to move wings to home position

We create a procedure (**home**) to move the wings and sliders back to their home positions, which is **45** for left wing and **135** for right wing.



```
when Button_Home .Click
do  call home
```



```
to home
do  call MT7697_LeftWingServo .Write
                             value 45
    call MT7697_RightWingServo .Write
                              value 135
    set Slider_LeftWing . ThumbPosition to 45
    set Slider_RightWing . ThumbPosition to 135
```

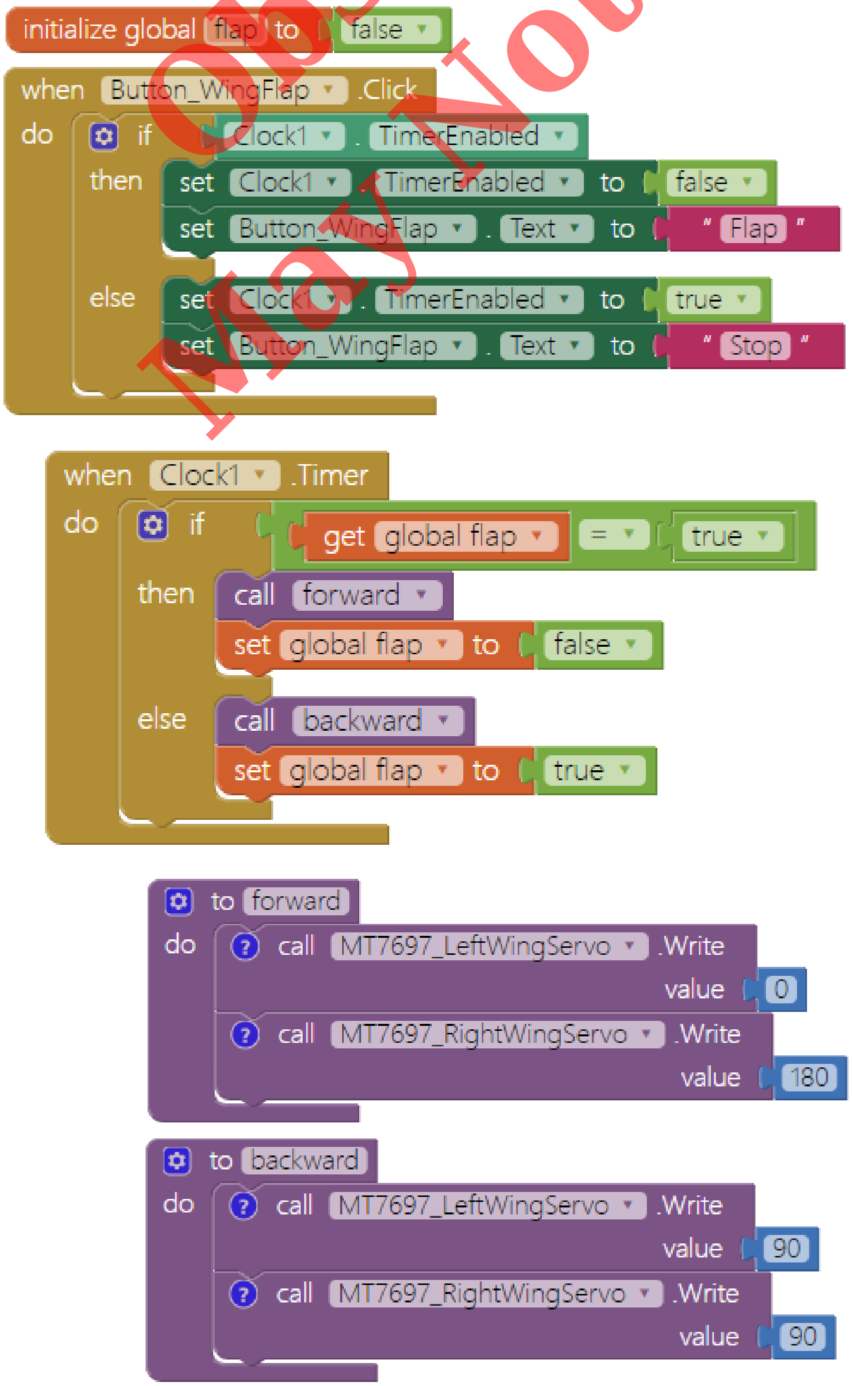
7. Button to flap wings

This section is exactly the same with Codi Bot wing tutorial.

We use **Button_WingFlap** to control the clock timer, which in turn makes the wings flapping.

For simplicity, we create two procedures (**forward**, **backward**) to control the servo motor positions.

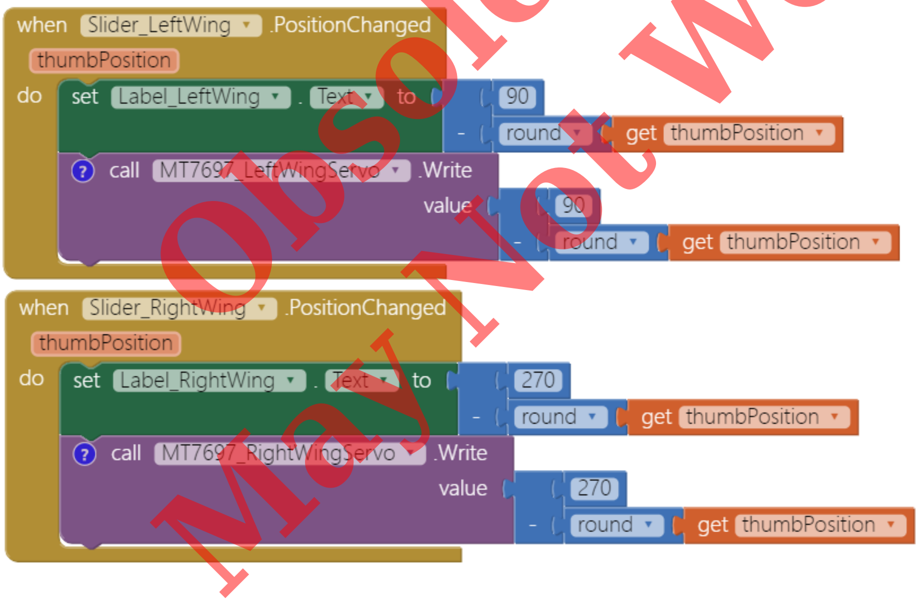
Obsolete
May Not Work



```
initialize global flap to false

when Button_WingFlap .Click
do  if Clock1 . TimerEnabled
    then set Clock1 . TimerEnabled to false
         set Button_WingFlap . Text to " Flap "
    else set Clock1 . TimerEnabled to true
         set Button_WingFlap . Text to " Stop "

when Clock1 .Timer
do  if get global flap = true
    then call forward
         set global flap to false
    else call backward
         set global flap to true

to forward
do  call MT7697_LeftWingServo .Write
                            value 0
    call MT7697_RightWingServo .Write
                             value 180

to backward
do  call MT7697_LeftWingServo .Write
                            value 90
    call MT7697_RightWingServo .Write
                             value 90
```

8. **Sliders to move each wing**

   When you drag a slider, the corresponding servo motor will move to the selected position.



```
when Slider_LeftWing .PositionChanged
  thumbPosition
do  set Label_LeftWing . Text to  90 - round get thumbPosition
    call MT7697_LeftWingServo .Write
                           value  90 - round get thumbPosition

when Slider_RightWing .PositionChanged
  thumbPosition
do  set Label_RightWing . Text to  270 - round get thumbPosition
    call MT7697_RightWingServo .Write
                            value  270 - round get thumbPosition
```

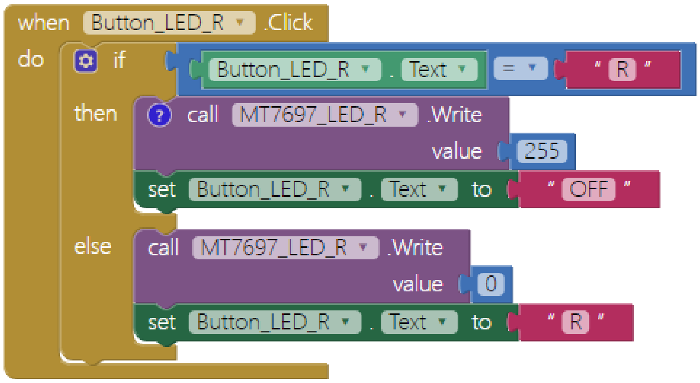
Obsolete
May Not Work

LED (Blocks)

Here we are going to control the LEDs with buttons and sliders, as we did in the [Codi Bot LED tutorial](#).

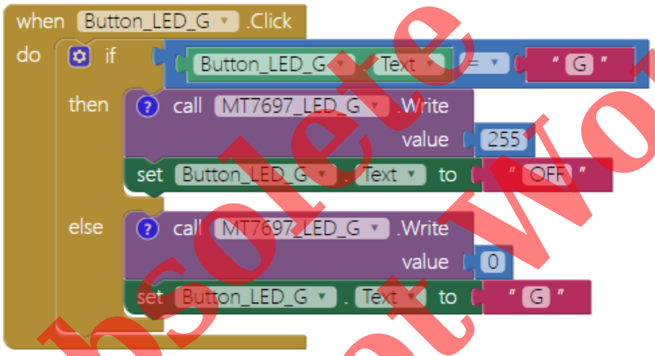
9. **Button to control RGB LED**

   These three buttons are used to control the red, green and blue colors of the RGB LED. For the red light, we use **MT7697_LED_R.Write** method. Value **255** turns on the red light; and value **0** turns it off. Please finish the other two button events in a same manner.

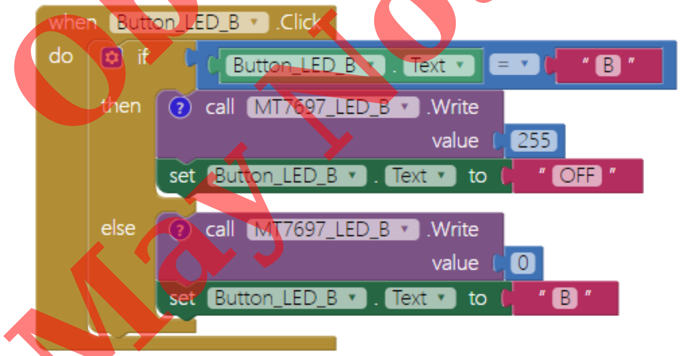


```
when Button_LED_R .Click
do  if    Button_LED_R . Text = " R "
    then  call MT7697_LED_R .Write
                         value  255
          set Button_LED_R . Text to " OFF "
    else  call MT7697_LED_R .Write
                         value  0
          set Button_LED_R . Text to " R "
```

Obsolete
May Not Work



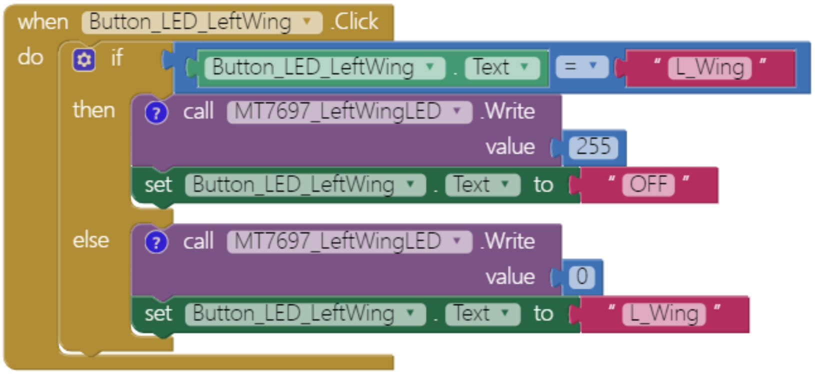
```
when Button_LED_G .Click
do  if  Button_LED_G . Text  =  " G "
    then  call MT7697_LED_G .Write
                          value  255
          set Button_LED_G . Text to " OFF "
    else  call MT7697_LED_G .Write
                          value  0
          set Button_LED_G . Text to " G "
```



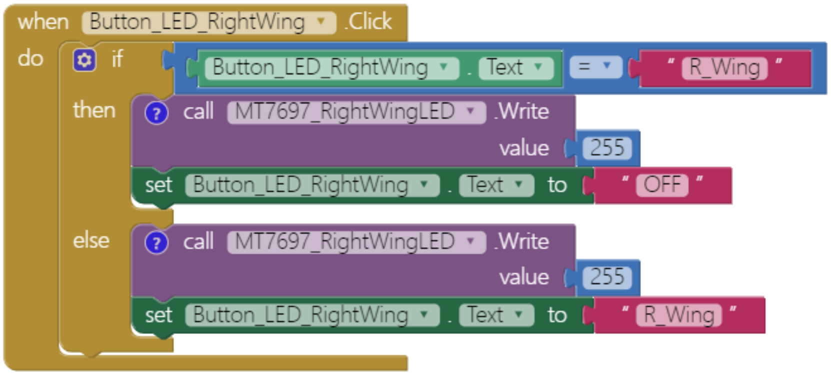
```
when Button_LED_B .Click
do  if  Button_LED_B . Text  =  " B "
    then  call MT7697_LED_B .Write
                          value  255
          set Button_LED_B . Text to " OFF "
    else  call MT7697_LED_B .Write
                          value  0
          set Button_LED_B . Text to " B "
```

10. Button to control the green and purple wing LEDs

These two buttons are used to control the LEDs of both wings. The idea is the same as in the previous step.



```
when Button_LED_LeftWing .Click
do  if  Button_LED_LeftWing . Text  =  " L_Wing "
    then  call MT7697_LeftWingLED .Write
                                value  255
          set Button_LED_LeftWing . Text to " OFF "
    else  call MT7697_LeftWingLED .Write
                                value  0
          set Button_LED_LeftWing . Text to " L_Wing "
```

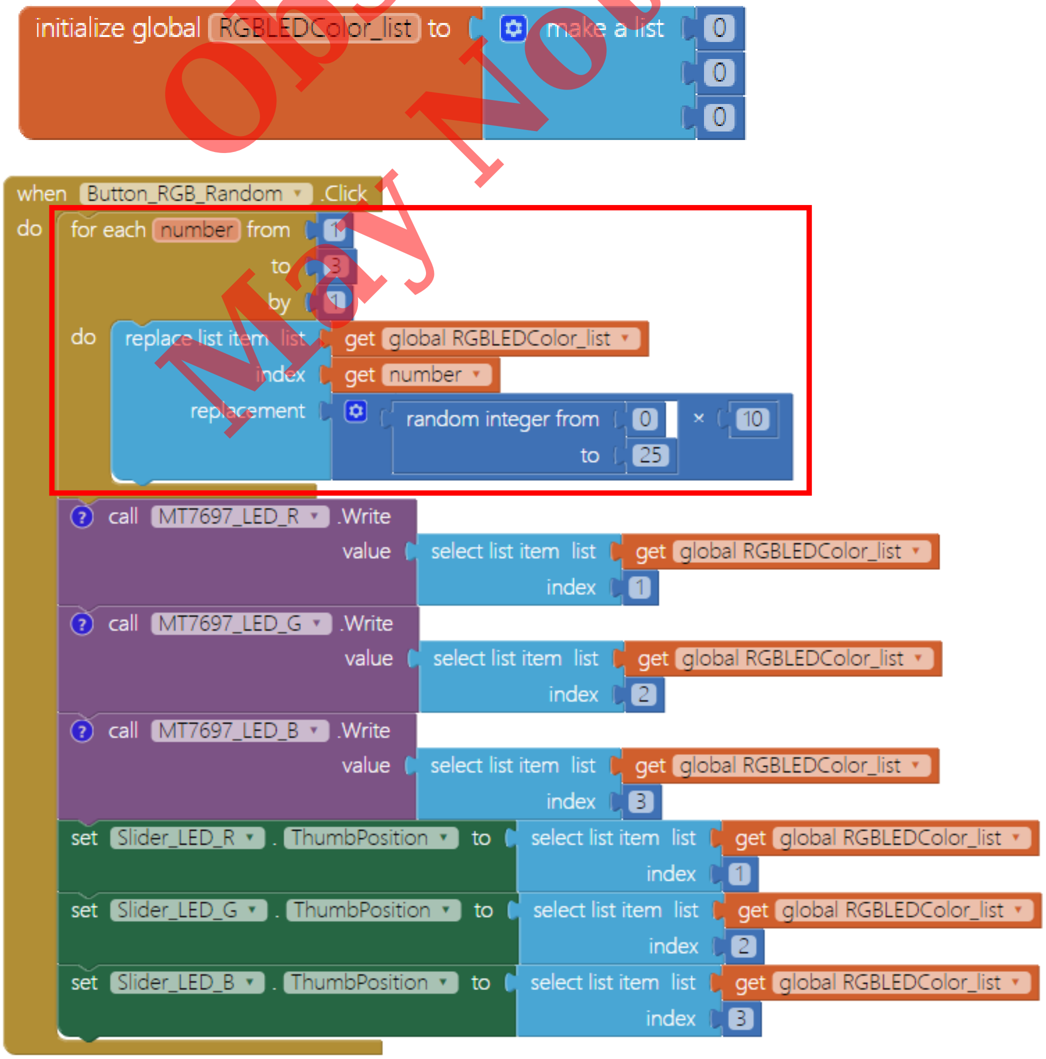


```
when Button_LED_RightWing .Click
do  if  Button_LED_RightWing . Text  =  " R_Wing "
    then  call MT7697_RightWingLED .Write
                                 value  255
          set Button_LED_RightWing . Text to " OFF "
    else  call MT7697_RightWingLED .Write
                                 value  255
          set Button_LED_RightWing . Text to " R_Wing "
```

11. Button to randomize the RGB LED color

We can also randomize the RGB LED color. When **Button_RGB_Random** is pressed, we use a loop (for each) to replace the **RGBLEDColor_list** variable content with a random number from **0** to **250**. Then we control the corresponding color and slider thumb position with this variable.

Obsolete
May Not Work

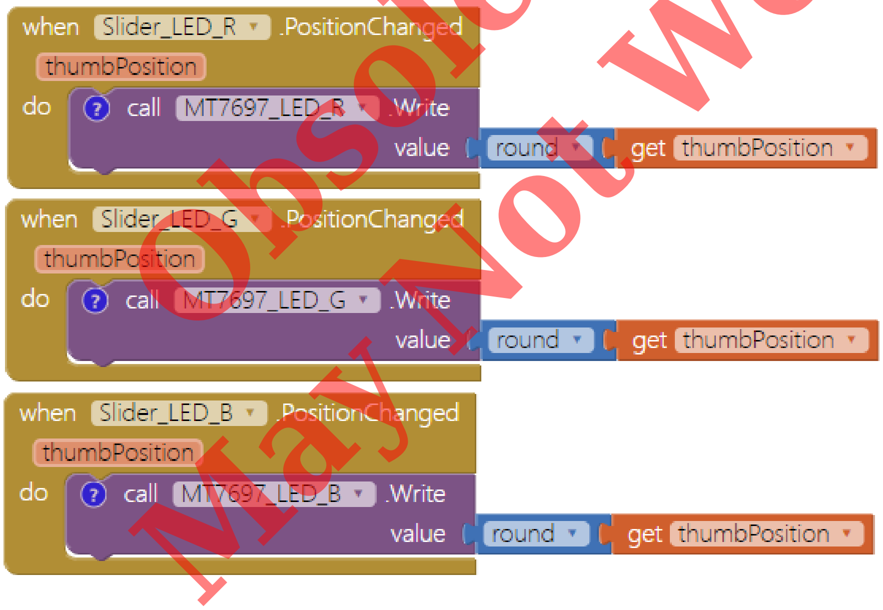


```
initialize global RGBLEDColor_list to  make a list  0
                                                    0
                                                    0

when Button_RGB_Random .Click
do  for each number from 1
                      to 3
                      by 1
    do  replace list item  list  get global RGBLEDColor_list
                           index get number
                     replacement random integer from 0 × 10
                                                  to 25
    call MT7697_LED_R .Write
                       value  select list item  list  get global RGBLEDColor_list
                                                index 1
    call MT7697_LED_G .Write
                       value  select list item  list  get global RGBLEDColor_list
                                                index 2
    call MT7697_LED_B .Write
                       value  select list item  list  get global RGBLEDColor_list
                                                index 3
    set Slider_LED_R . ThumbPosition to  select list item  list  get global RGBLEDColor_list
                                                           index 1
    set Slider_LED_G . ThumbPosition to  select list item  list  get global RGBLEDColor_list
                                                           index 2
    set Slider_LED_B . ThumbPosition to  select list item  list  get global RGBLEDColor_list
                                                           index 3
```

12. Sliders to fine tune the RGB LED color

When the slider is dragged, we control the corresponding color by slider thumb position.

Obsolete
May Not Work



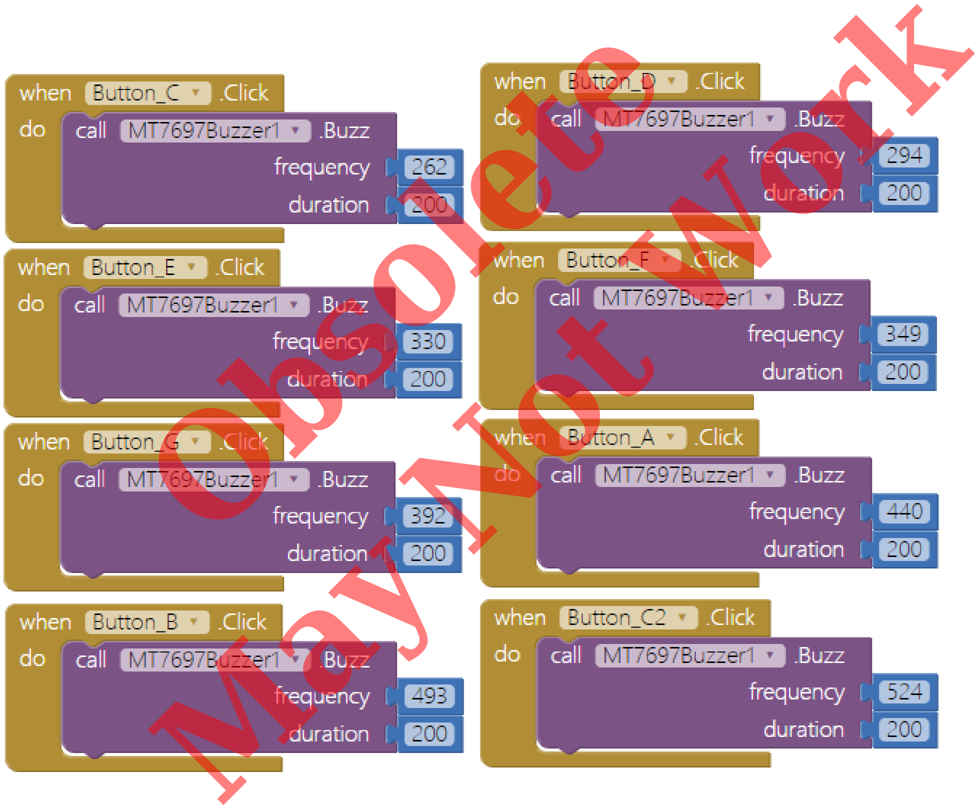
```
when Slider_LED_R .PositionChanged
  thumbPosition
do  call MT7697_LED_R .Write
                    value  round  get thumbPosition

when Slider_LED_G .PositionChanged
  thumbPosition
do  call MT7697_LED_G .Write
                    value  round  get thumbPosition

when Slider_LED_B .PositionChanged
  thumbPosition
do  call MT7697_LED_B .Write
                    value  round  get thumbPosition
```

Sound (Blocks)

Here we can play different notes with buttons, which are exactly the same with the **Codi Bot Sound tutorial**.

13. Buttons to make sounds of different notes

We have eight buttons here. Each of them will tell the buzzer inside the Codi Bot to make a different note. For example, when **Button_C** is pressed (**Button_C.Click** event), we use a **MT7697Buzzer.Buzz** method to make a sound of Note C by specifying the **frequency** as 262 and the **duration** as 200 (milliseconds). Please finish the other seven button events in the same manner.



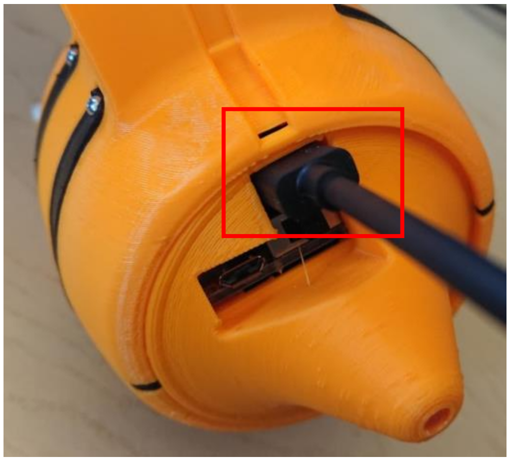
Obsolete May Not Work

```
when Button_C .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 262
                  duration  200

when Button_D .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 294
                  duration  200

when Button_E .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 330
                  duration  200

when Button_F .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 349
                  duration  200

when Button_G .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 392
                  duration  200

when Button_A .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 440
                  duration  200

when Button_B .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 493
                  duration  200

when Button_C2 .Click
do   call MT7697Buzzer1 .Buzz
                  frequency 524
                  duration  200
```

Arduino IDE and sketch

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

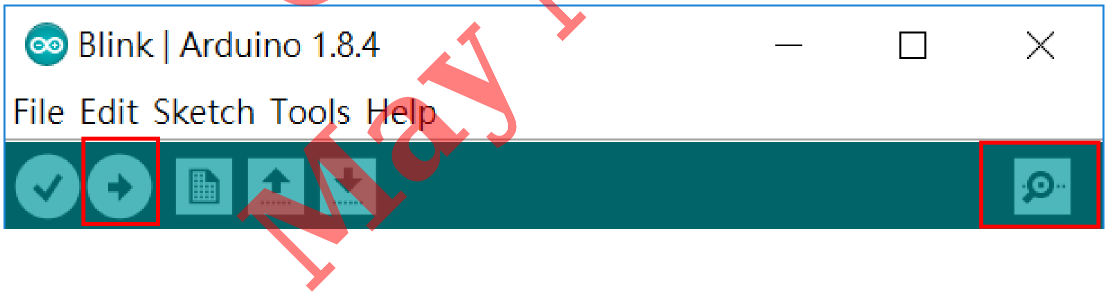
Connect your computer and LinkIt 7697 with a microUSB cable.

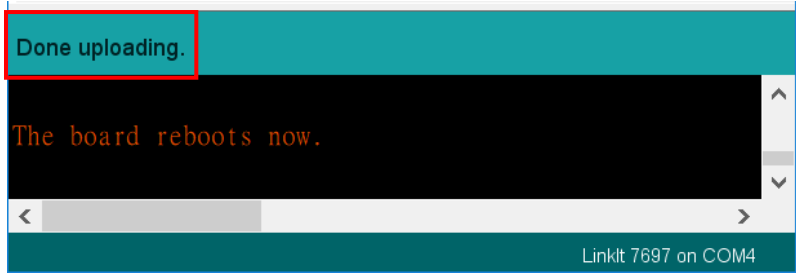


Obsolete May Not Work

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

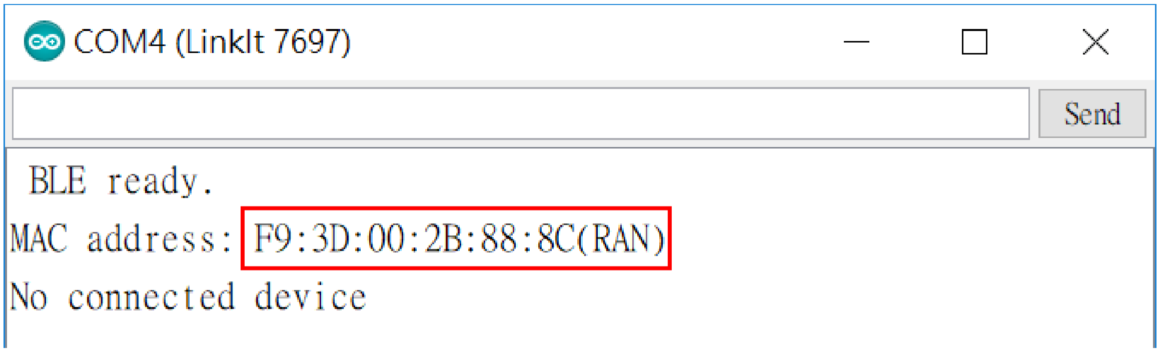
Press the "**Upload**" right-arrow button of Arduino IDE. This will compile and upload the Arduino sketch to your LinkIt 7697. Please make sure you can see the "**done uploading**" message in the console before continuing.




Done uploading.

```
The board reboots now.
```

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 will choose the device of this address in our app.

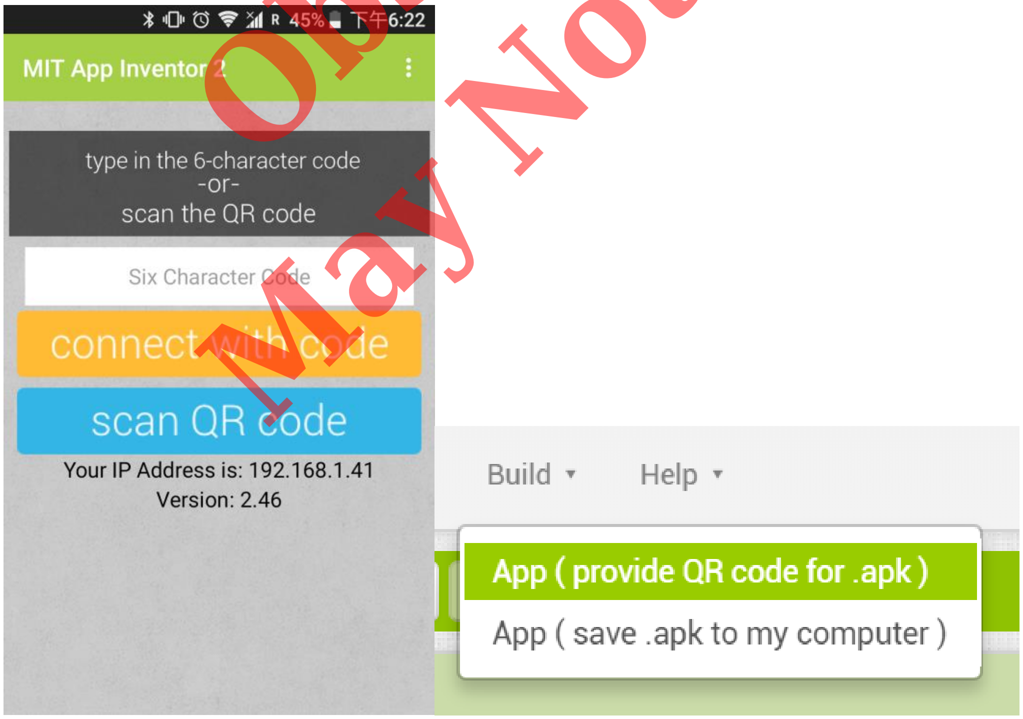


```
 BLE ready.
MAC address: F9:3D:00:2B:88:8C(RAN)
No connected device
```

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 QR code for the .apk file of this project. Use MIT AI2 Companion to scan this QR code, download the app, and install it.

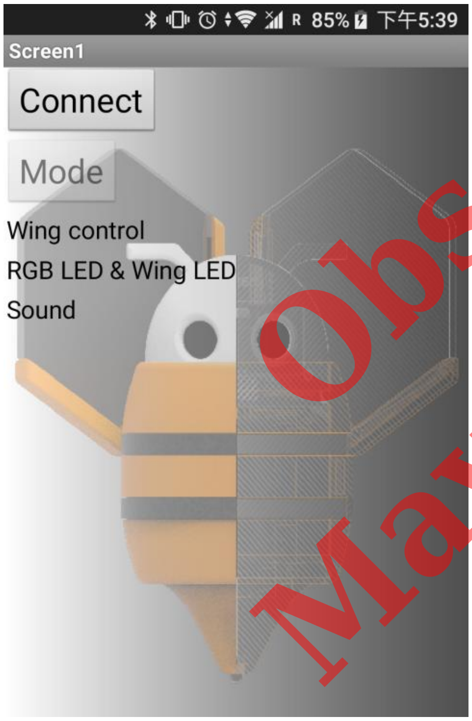
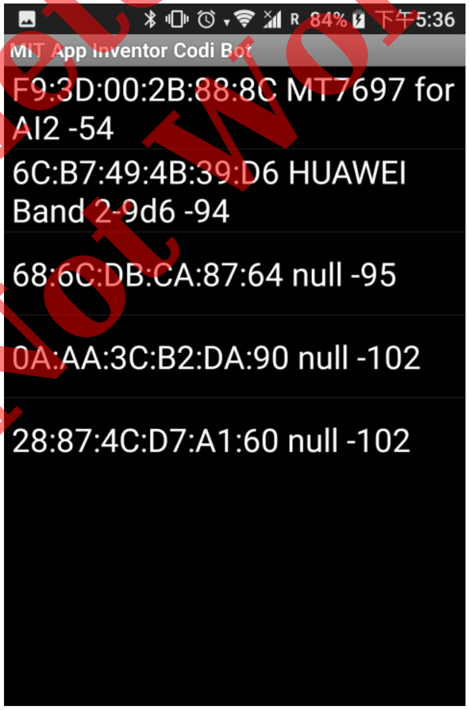
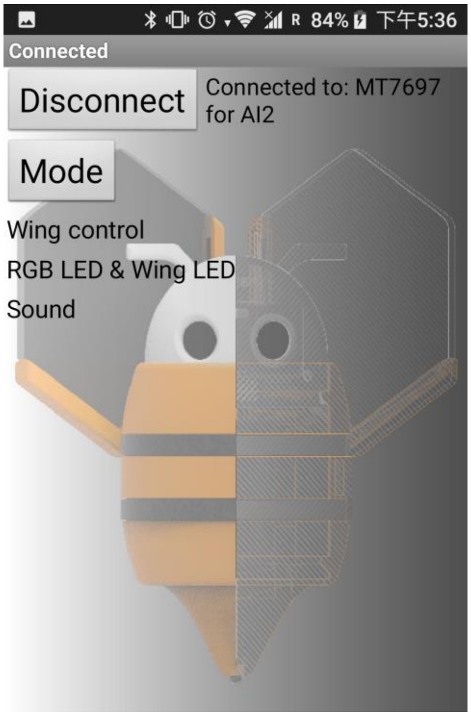
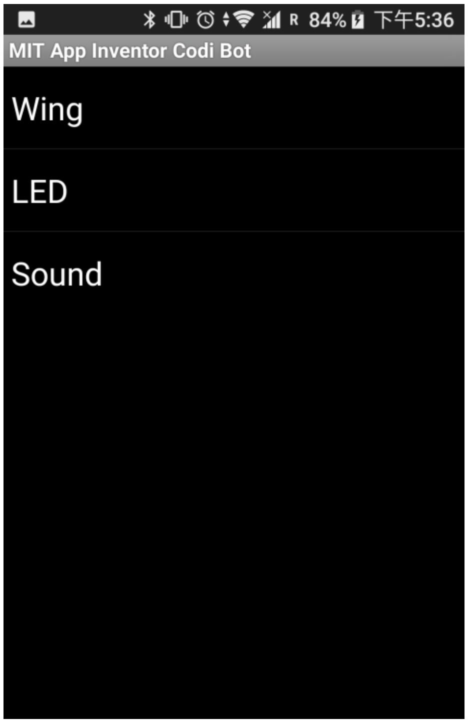
Obsolete
May Not Work



Your IP Address is: 192.168.1.41
Version: 2.46

Open Codi Bot app (Fig 1.) and click **Connect** ListPicker. You'll see a list of available Bluetooth devices (Fig 2). In general, the first item is the Codi Bot where "**F9:3D:00:2B:88:8C**" is the Bluetooth address of my Codi Bot. Click it and your phone will try to connect with the Codi Bot, If it connects successfully, you will see your app like in Fig 3.

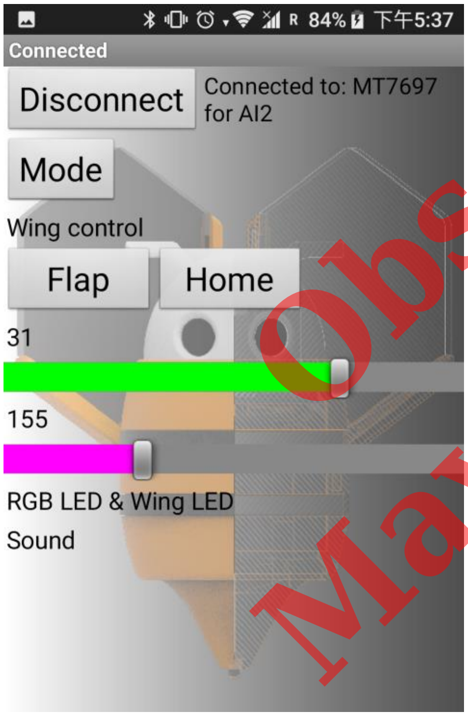
Click **Mode** ListPicker to choose which mode you want to play with: **Wing**, **LED** or **Sound** (Fig. 4). Choose one to interact with your Codi Bot (Fig 5 to Fig 7). Remember to click the **Disconnect** button when you finish with this project.

Fig 1. Initial screen	Fig 2. Select available Bluetooth device
 85% 下午5:39 Screen1 Connect Mode Wing control RGB LED & Wing LED Sound	 F9:3D:00:2B:88:8C MT7697 for AI2 -54 6C:B7:49:4B:39:D6 HUAWEI Band 2-9d6 -94 68:6C:DB:CA:87:64 null -95 0A:AA:3C:B2:DA:90 null -102 28:87:4C:D7:A1:60 null -102
Fig 3. Connected to Codi Bot (MT7697 for AI2)	Fig 4. Select mode
 84% 下午5:36 Connected Disconnect Mode Wing control RGB LED & Wing LED Sound Connected to: MT7697 for AI2	 84% 下午5:36 MIT App Inventor Codi Bot Wing LED Sound

Obsolete
May Not Work

Obsolete
May Not Work

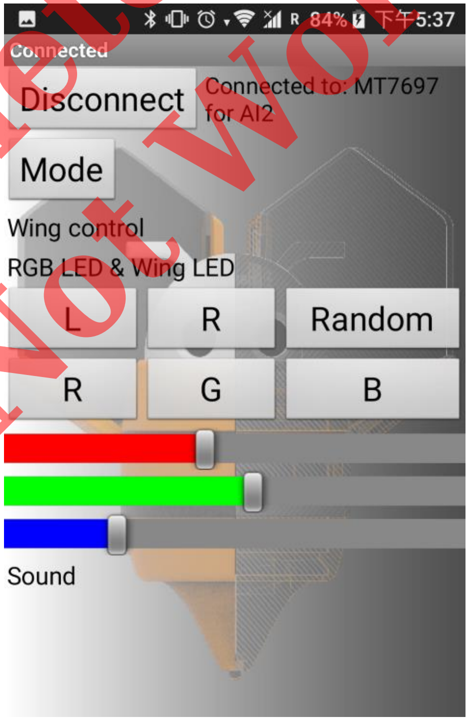
Fig 5. Wing mode



Connected

Connected to: MT7697 for AI2

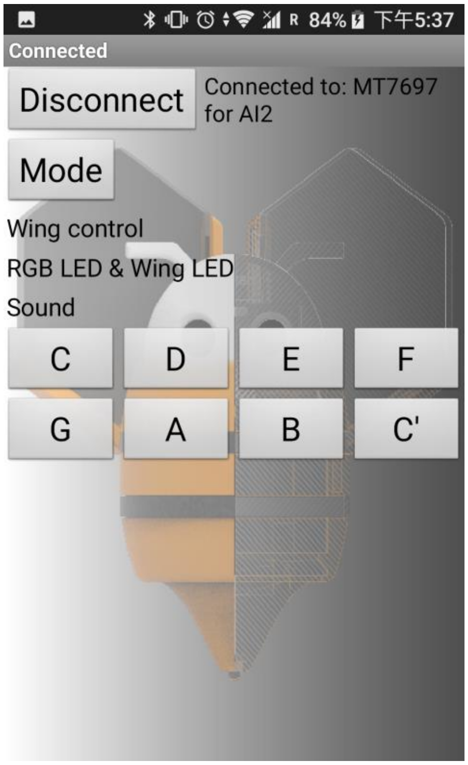
Fig 6. LED mode



Connected

Connected to: MT7697 for AI2

Fig 7. Sound mode



Connected

Connected to: MT7697 for AI2