App Inventor + IoT: Bouncing ball with Micro:bit buttons

60 min

(with Basic Connection tutorial completed)

Level: advanced

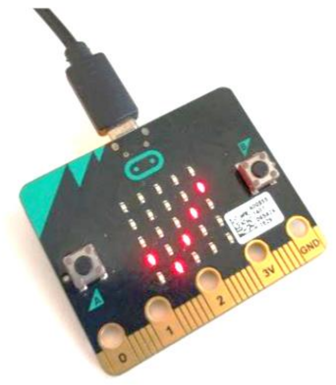
This tutorial will help you get started with App Inventor + IoT and the **BBC micro:bit** controller. Press the two buttons on micro:bit will make a ball on your app moving back and forth.

Hardware

You only need one **BBC micro:bit** to get started with this project.

Paring with Micro:bit

First, you will need to pair your Android phone or tablet to the micro:bit controller, using these directions. Your device must be paired with the micro:bit in order for the app to work.



App Inventor

This app can let you control a ball in the app to move back and forth by pressing two Micro:bit buttons. Now log into **MIT App Inventor site** and create a new project.

You should complete the **App Inventor + IoT Basic Connection tutorial** to make a basic connection to the micro:bit device. If you prefer, you can download the completed .aia file **here**.

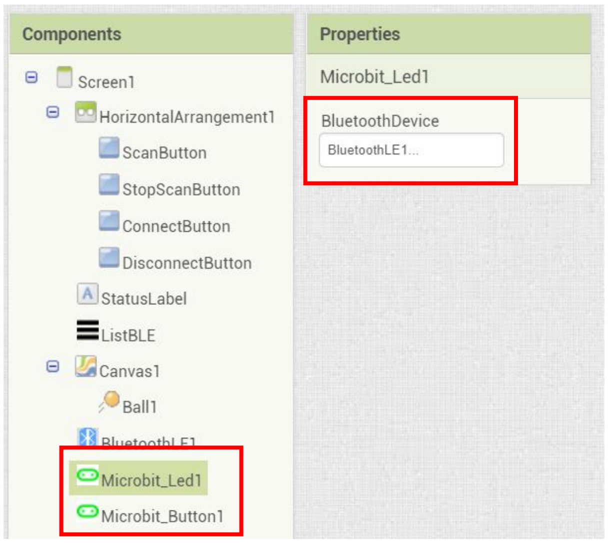
- App Inventor's micro:bit button component's document

The remaining steps all build off of the the starter code for Basic Connection tutorial and .aia.

Designer

First, we need to add the necessary extension.

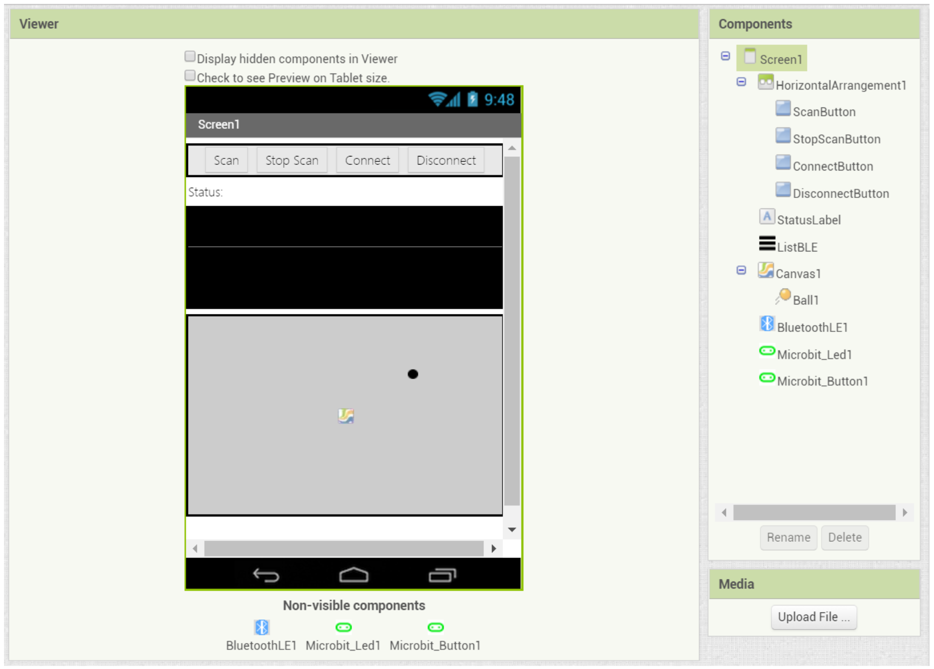
- In the Palette window, click on Extension at the bottom and then on "Import extension" and click on "URL".
  - Paste in this URL: http://iot.appinventor.mit.edu/assets/com.bbc.micro:bit.profile.aix
- Add a **Microbit_Buttons** extension to your app by dragging it onto the Viewer, set its **BluetoothDevice** property to "BluetoothLE1".
- Add a **Microbit_Led** extension, also set its **BluetoothDevice** property to "BluetoothLE1".



Let’s add more components to our app to receive the micro:bit buttons’ statuses.

- From the Drawing and animation drawer in the Palette, drag in a **Canvas** and a **Ball**. Set Canvas’s height to 320 pixels, width to fill parent (or any parameters you like).

Your designer page should look like this:

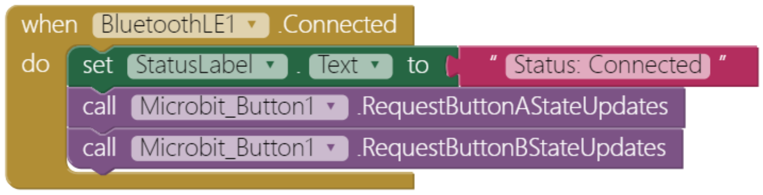


Blocks

We want to control Ball component's horizontal movement with the two buttons on out micro:bit controller. Let's begin:

STEP1: Request updates when connected

In the **BluetoothLE1.Connected** event, we show connection status on label and request that the micro:bit update the two buttons' statuses.

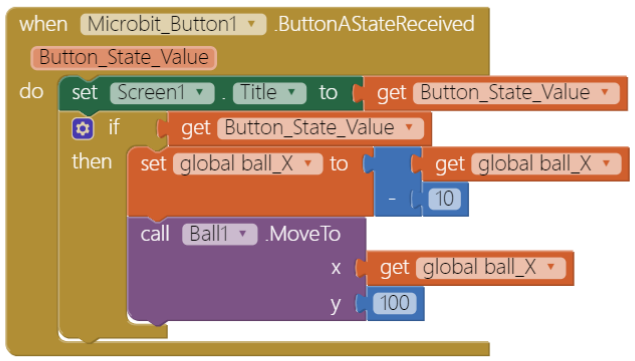


```
when BluetoothLE1 .Connected
do  set StatusLabel . Text to " Status: Connected "
    call Microbit_Button1 .RequestButtonAStateUpdates
    call Microbit_Button1 .RequestButtonBStateUpdates
```

STEP2: Micro:bit's A button pressed

In **Microbit_Button1.ButtonAStateReceived** event:

- If A button is pressed(**Button_State_Value** is true), then we set the **ball_X** variable to decrease by 10.
- Make Ball1 component move to position (ball_X, 100) to make it move left by 10 pixels.



```
when Microbit_Button1 .ButtonAStateReceived
  Button_State_Value
do  set Screen1 . Title to get Button_State_Value
    if   get Button_State_Value
    then set global ball_X to  get global ball_X - 10
         call Ball1 .MoveTo
                      x  get global ball_X
                      y  100
```

STEP3: Micro:bit's B Button pressed

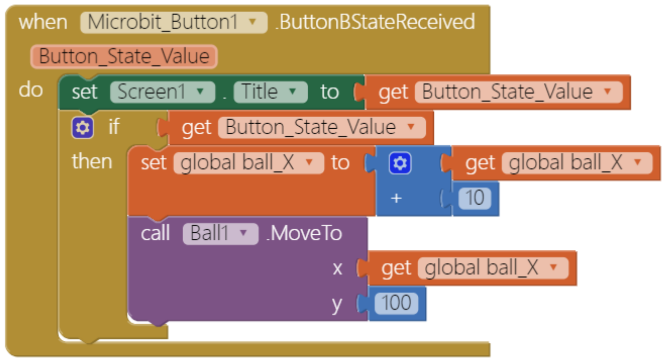
For the **Micro:bit's B Button**, things are almost the same, just in the opposite direction.

In **Microbit_Button1.ButtonBStateReceived** event:

- If user pressed Micro:bit's A button is pressed

(**Button_State_Value** will be **true**), then we set **ball_X** variable increase by10.

- Make Ball1 component move to position (ball_X, 100) to make it move left by 10 pixels.

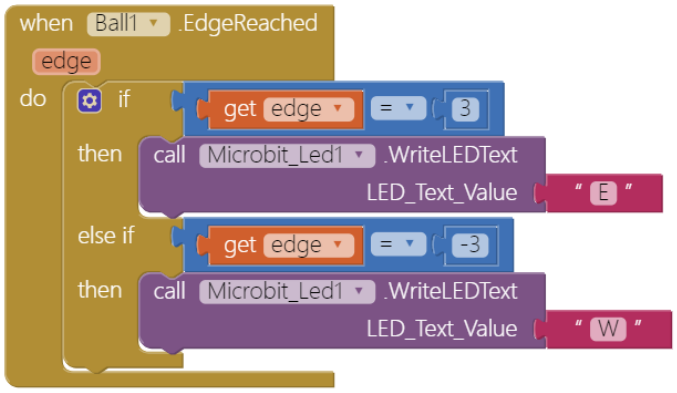


```
when Microbit_Button1 .ButtonBStateReceived
  Button_State_Value
do  set Screen1 . Title to  get Button_State_Value
    if   get Button_State_Value
    then set global ball_X to  get global ball_X + 10
         call Ball1 .MoveTo
                        x  get global ball_X
                        y  100
```

STEP4 **Ball reached canvas' edge**

When Ball1 reached canvas' edges (**Ball1.EdgeReached** event), we will show corresponding text (single character) on micro:bit's Led matrix.

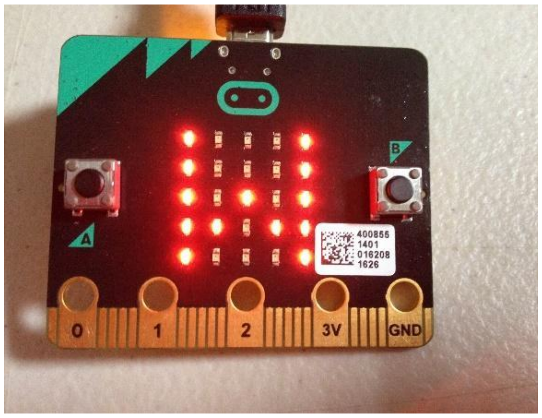
In **Ball1.EdgeReached** event, we use the if/else block if to check which edge is reached, then send '**E**' or '**W**' character (meaning **East** or **West**) to the micro:bit using the **Microbit_Led1.WriteLEDText** method.

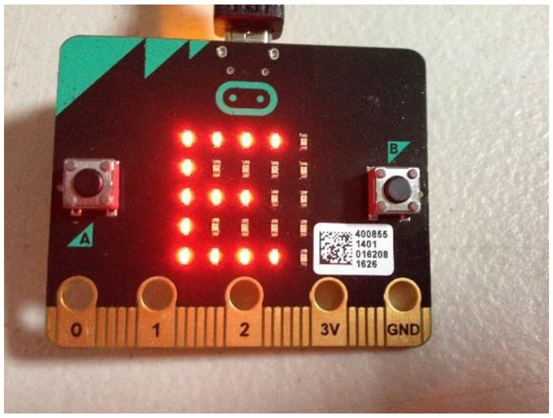


```
when Ball1 .EdgeReached
  edge
do  if      get edge = 3
    then    call Microbit_Led1 .WriteLEDText
                         LED_Text_Value  " E "
    else if get edge = -3
    then    call Microbit_Led1 .WriteLEDText
                         LED_Text_Value  " W "
```

Tips

Your app should now be working! Pair the Bluetooth on your Android device to test it out! Connect your micro:bit device using the MIT AI2 Companion (if you haven't already) or install the .apk. Press the two buttons on micro:bit, and the ball on the screen should move left and right.





Brainstorming

1. Try to move a Micro:bit LED dot back and forth by Micro:bit buttons (refer to our Micro:bit LED tutorial).