App Inventor + IoT: Control RGB LED with Micro:bit I/O pins

60 mins

(with Basic Connection tutorial completed)

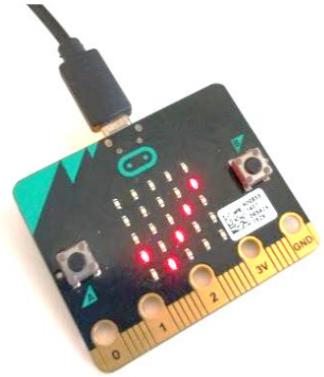
Level: advanced

This tutorial will help you work with App Inventor + IoT and control a RGB LED on a micro:bit controller.

- source .aia

Pairing with Micro:bit

First, you will need to pair your 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.

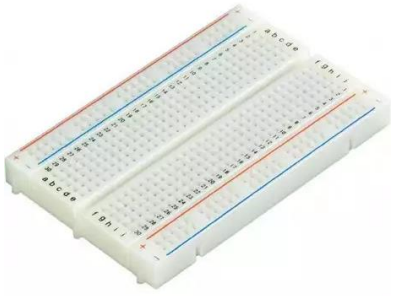


Hardware list

In this project, we are going to control a RGB LED (which is connected to Micro:bit) using App Inventor. RGB means this knd of LED actually has three different color LEDs inside (red, green and blue).

Here are the components you need for this project:

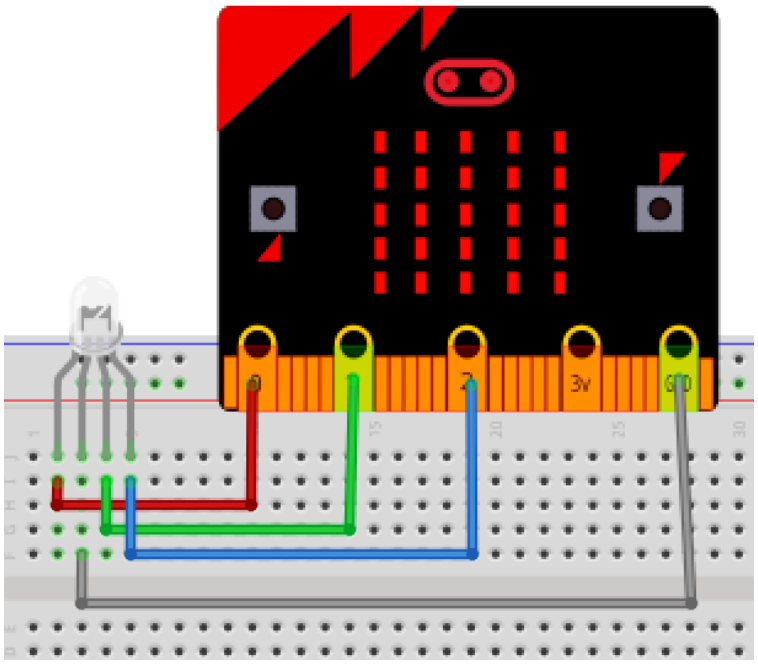
- BBC micro:bit dev board, 1
- breadboard, 1
- wires, 4
- RGB LED (common cathode), 1




Demo video: https://youtu.be/TQcRy1JkFBc

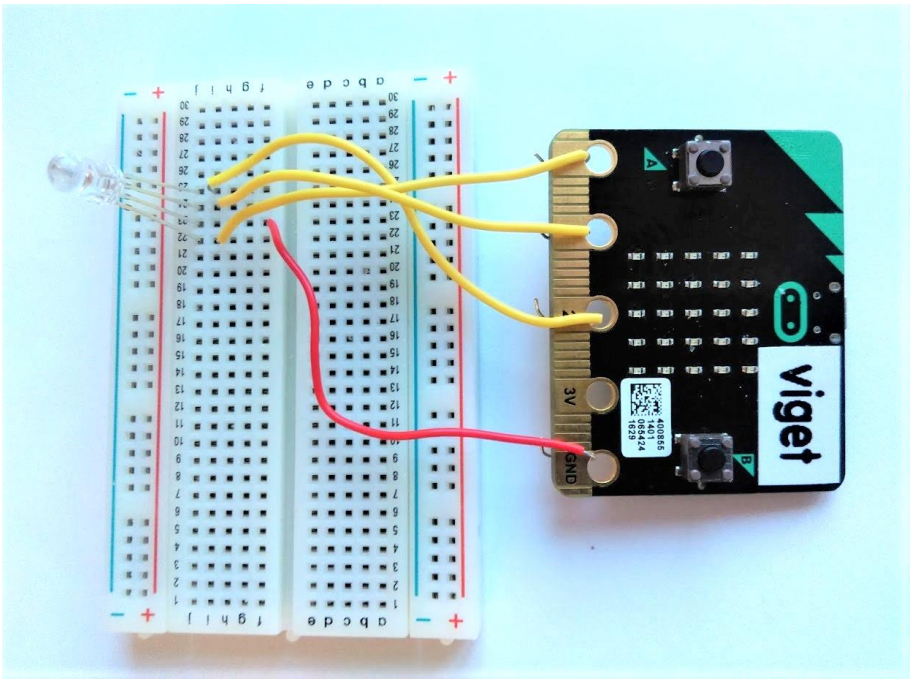
Please connect Micro:bit and RGB LED according to this table:

Micro:bit	**RGB LED (common cathode)**
GND	GND (longest pin, **grey** wire)
P0	R (**red** wire)
P1	G (**green** wire)
P2	B (**blue** wire)



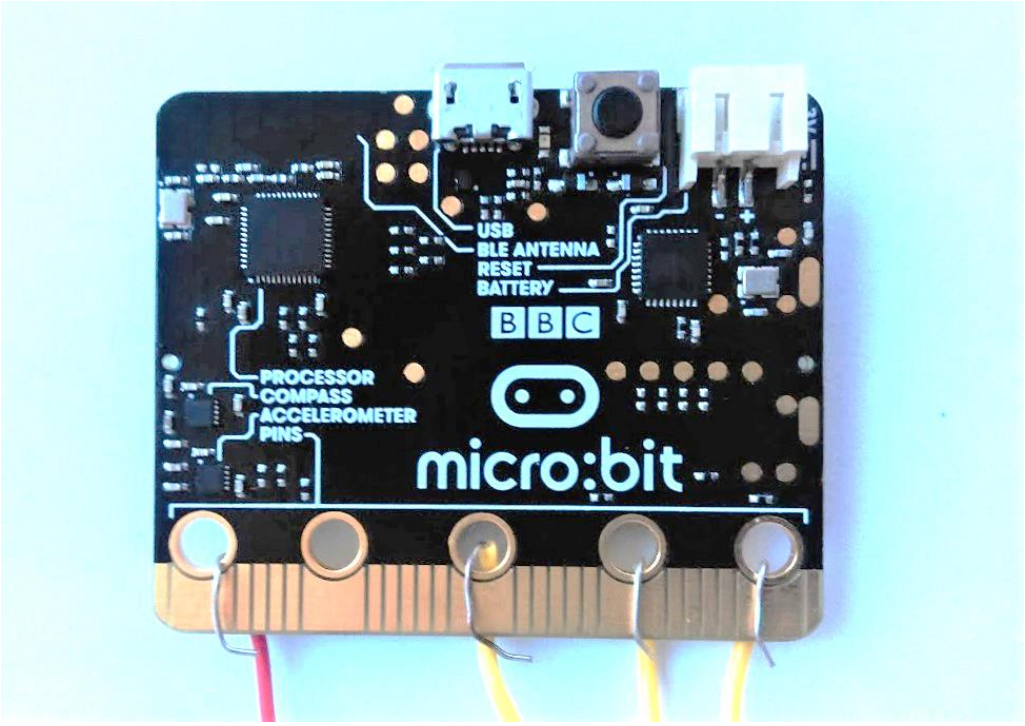
Finish as below, let's take a look:

Blue
Green
GND
Red



P0
P1
P2
GND

Bend the wire to U shape and hook on Micro:bit pins.



GND	2	1	0

App Inventor

This app lets you make an RGB LED light up in four different colors by clicking buttons on your app. First, log into **MIT App Inventor site** and create a new project.

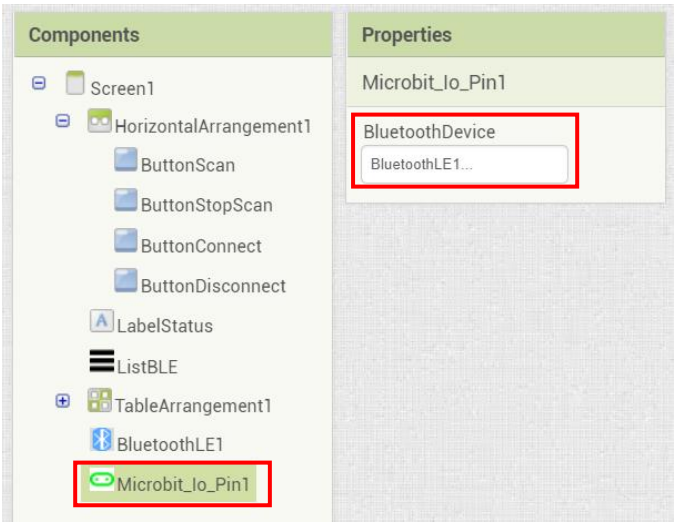
Designer

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**.

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

First, we need to add the necessary extension.

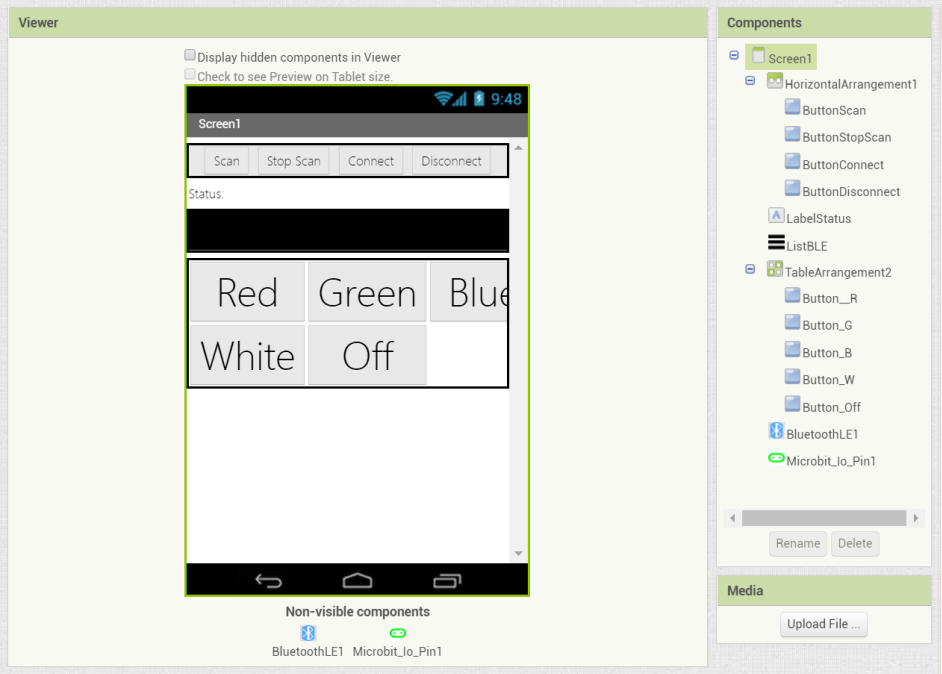
- In the Palette window, click on Extension at the bottom and then on "**Import extension**" and click on "**URL**".
    - Paste micro:bit extension URL: http://iot.appinventor.mit.edu/assets/com.bbc.microbit.profile.aix
- Add a **Microbit_IOpin** component to your app by dragging it onto the Viewer, set its *BluetoothDevice* to "**BluetoothLE1**" (Don't forget!).



Let’s add more components into our app to control micro:bit’s I/O pins.

- From the Layout palette, drag in a **TableArrangement** component.
    - Set its width to "**Fill parent**", height to **200** pixels, row to **2** and column to **3**.
    - Set its **Visible** property to **false**, it will be set to **true** after the Bluetooth connection between micro:bit is established.
- Add five buttons into the tablearrangement component, and set their text properties to "**Red**", "**Green**", "**Blue**", "**White**" and "**Off**", representing different colors of the RGB LED light.

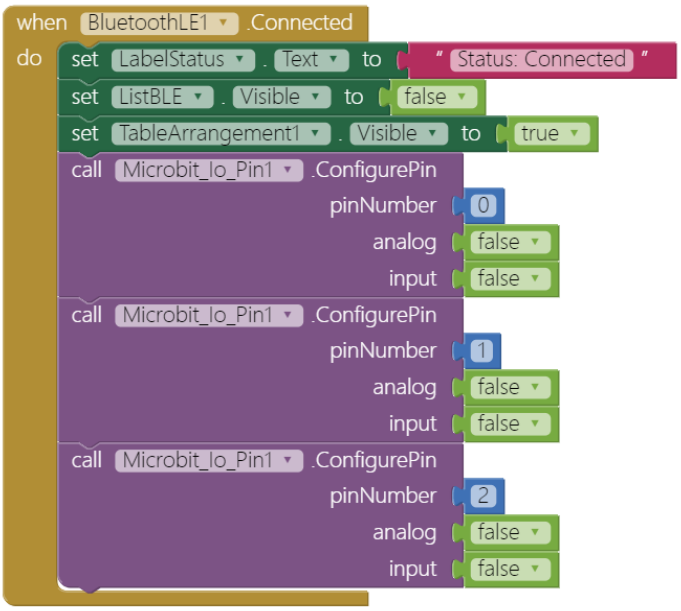
After some adjusting, your designer should look similar to this. It doesn’t have to be exactly the same. Feel free to modify the component properties, such as background color, position and text size.



Blocks

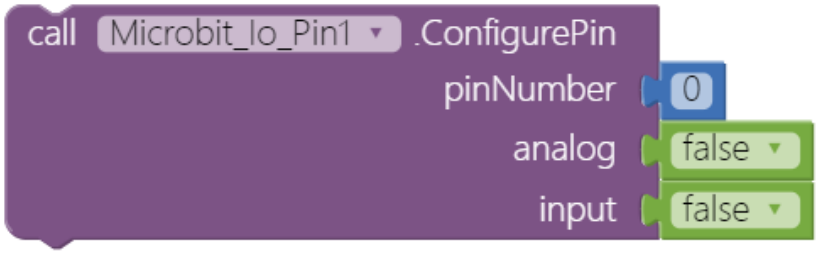
STEP 1: Request updates when connected

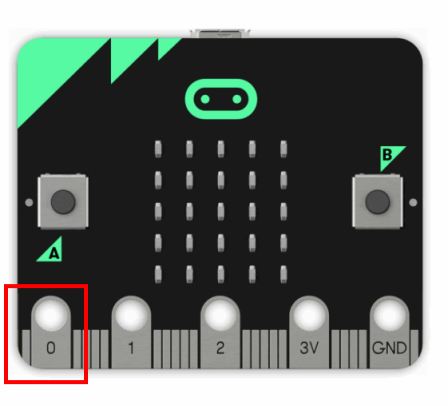
In the **BluetoothLE1.Connected** event, we show messages to tell user that we are connected with micro:bit and set micro:bit's pin status as "**digital output**" since we are going to control a RGB LED in this project. With three **Microbit_Io_Pin.ConfigurePin** methods, please specify the **pinNumber** to 0, 1 and 2 (means P0, P1 and P2 pin of micro:bit and set the **analog** field to **false** and the input field to **false**.



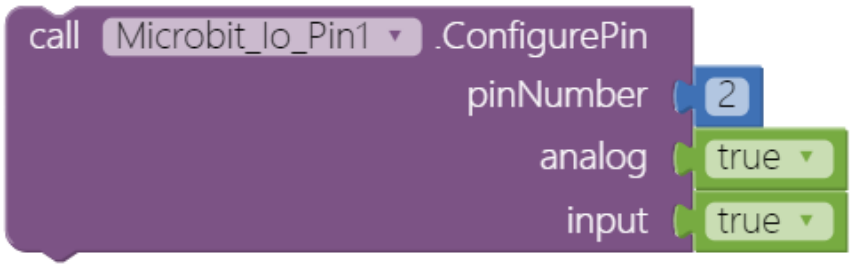
Let's get a look at **Microbit_Io_Pin.ConfigurePin** method. It has three parameters: **pinNumber (pin index)**, **analog** (true to analog, false to digital) and **input** (true to input, false to output).

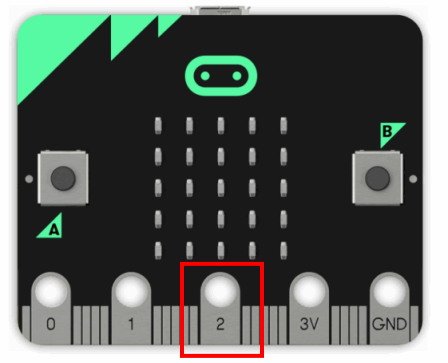
This is to set micro:bit **P0** pin as **digital output**. You can connect component like LED or relay module to this pin. For micro:bit I/O pins detail please check this link: http://microbit.org/guide/hardware/pins/





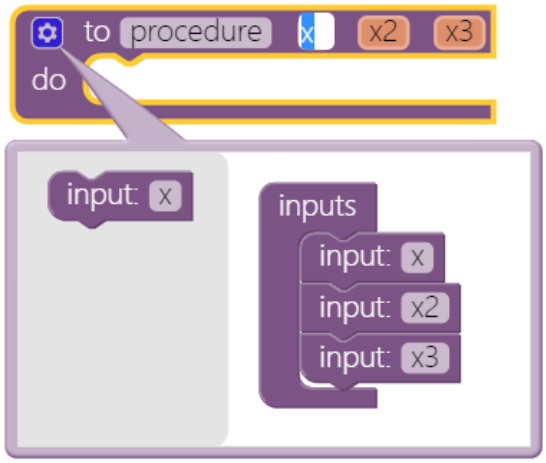
And this sets micro:bit's **P2** pin as **analog input**. You can connect a component like a potentiometer to this pin.



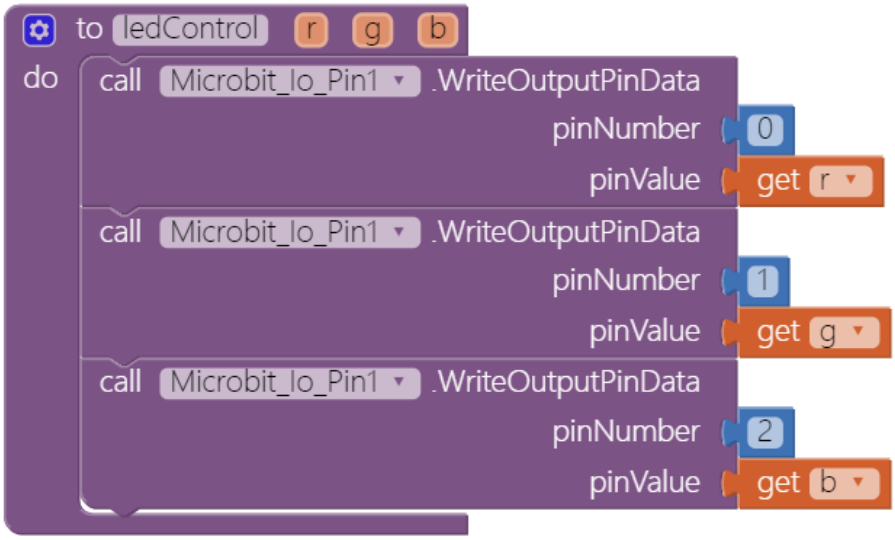


STEP2: Procedure to update LED status

Here we use a procedure to manage instructions for control micro:bit pin status. Please add a procedure and click the blue gear to add three parameters. Rename this procedure to "**ledControl**", and add three parameters as "**r**", "**g**" and "**b.**" Each of them is used to control a pin of RGB LED.



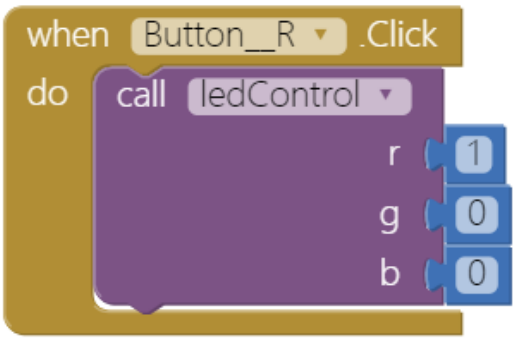
Now add three **Microbit_Io_Pin.WriteOutputPinData** methods into this procedure. For the first one, since we've connected RGB LED's **R/G/B** pin to micro:bit's **P0/P1/P2** pin, we set the first method's pinNumber to **0** and pinValue to **r** variable; the second one's to **1** and **g** variable and the third one's to **2** and **b** variable. Our ledControl procedure is finished as below:



STEP2: Button to light up red

When **Button_R** is pressed, we call **ledControl** procedure in the previous step and set its parameters to (1, 0, 0), this means light the LED in red color (by putting **P0** to high voltage level and **P1/P2** to

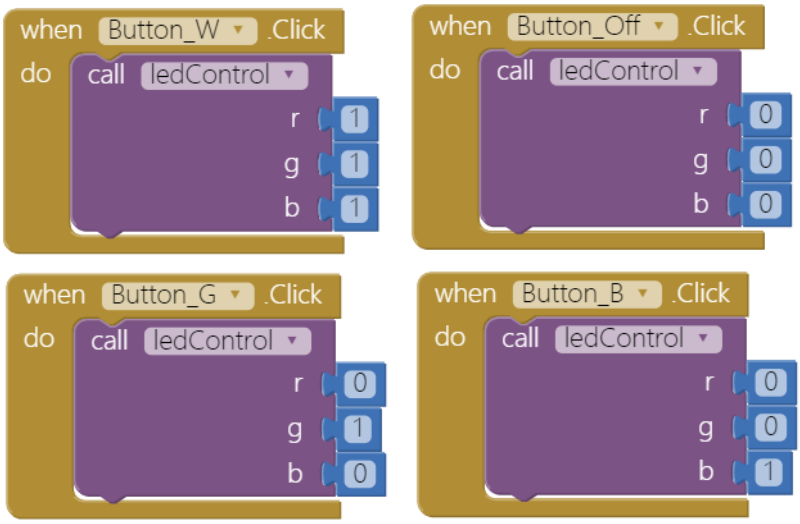
low).



STEP3: other buttons for different colors

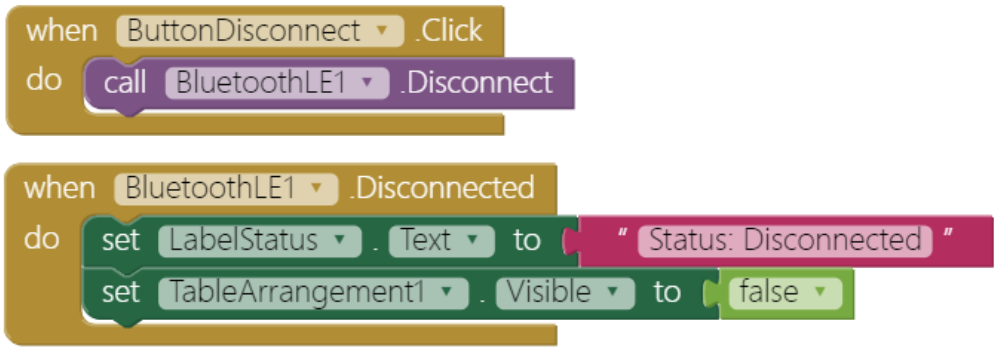
For the other four buttons, we still call **ledControl** procedure but with different parameters:

- **Button_G**: (0, 1, 0) to light up in green color.
- **Button_B**: (0, 0, 1) to light up in blue color.
- **Button_W**: (1, 1, 1) to light up in white color.
- **Button_Off**: (0, 0, 0) to turn the light off.



STEP4: Disconnect from the micro:bit

You can disconnect from the micro:bit by clicking the **ButtonDisconnect**. This will reset the app to its initial state to wait for next connect request.



Tips

Your app should now be working! Make sure you have paired the Bluetooth on your Android device to your micro:bit. Then test it out by connecting your micro:bit device using the MIT AI2 Companion (if you haven't already) or installing it by .apk.
Press buttons on the app to see if our RGB LED lights up with the correct color.

Brainstorming

1. Try to light up the RGB LED with different colors, such as red and blue lighting up together to shine in purple.
2. Add a **SpeechRecognizer** component to control the RGB LED by your voice command.