App Inventor + IoT:
Read soil moisture status with LinkIt 7697

90
min

Level: advanced

This tutorial will help you get started with App Inventor + IoT. We are going to monitor the moisture level of your plant soil with a LinkIt 7697 (Arduino compatible) dev board. Here we are using **Seeed Studio's soil moisture sensor**, but any other moisture sensor or analog input component, such as a potentiometer, photoresistor, or thermo-resistor, should be fine.

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

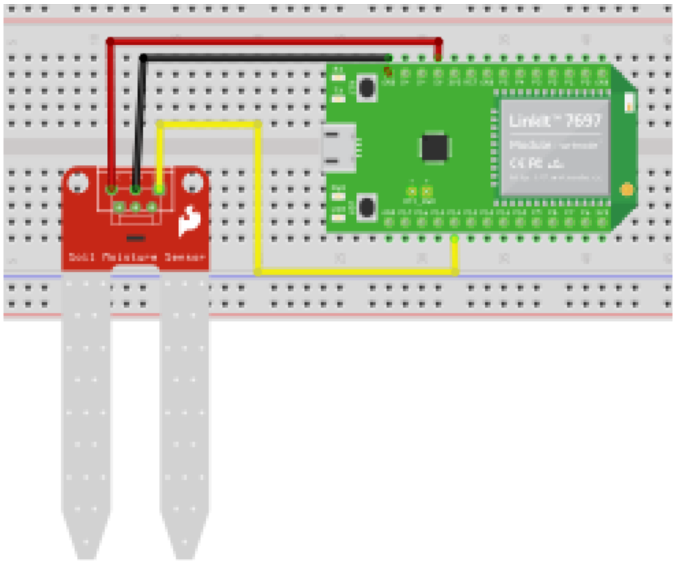
Hardware

LinkIt 7697 is an Arduino compatible dev board with Wi-Fi / BLE. You can use it to interface with App Inventor through BLE communication.

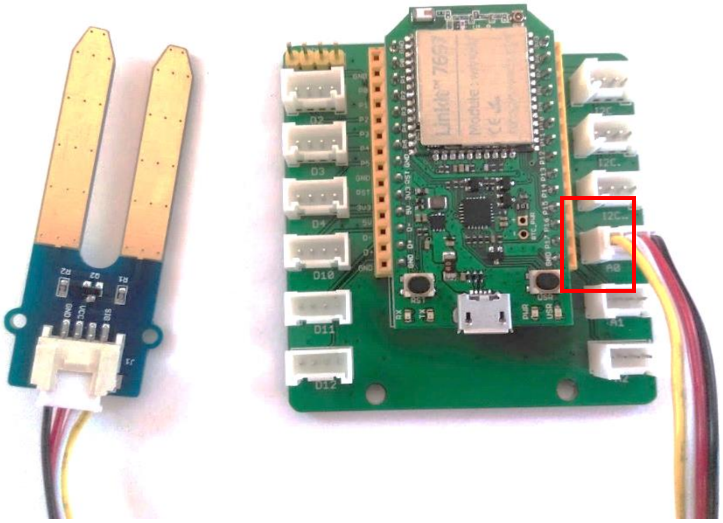
In this project, we are going to send moisture sensor data (connected to LinkIt 7697) to App Inventor. Here are the components we need to build this project:

- LinkIt 7697 dev board, 1
- breadboard, 1
- wires, 3
- Any 3-pin analog moisture sensor, 1
- ***Grove soil moisture sensor**
- ***Grove extension shield for LinkIt 7697**

If you use a general 3-pin analog sensor, please connect the SIG pin to the LinkIt 7697's P14(A0), Vcc to 5V, GND to GND. This is what it should look like when you are finished:

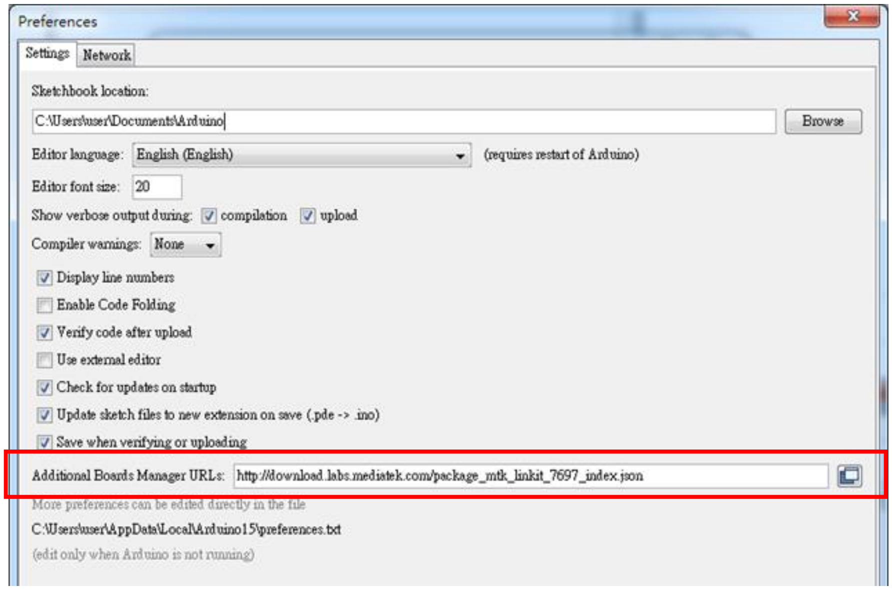


Or you can use Seeed Studio’s **Grove extension shield for LinkIt 7697**, just connect **Grove soil moisture sensor** to the shield’s A0 port, like below:



Arduino IDE Setup

1. First get Arduino IDE 1.8.x version, download the .zip file, unzip and click arduino.exe to open the IDE.
2. From the **File/ Preference** menu, enter the link below to the Additional Boards Manager URLs field:
   - http://download.labs.mediatek.com/package_mtk_linkit_7697_index.json



Sketchbook location: C:\Users\user\Documents\Arduino

Editor language: English (English) (requires restart of Arduino)

Editor font size: 20

Show verbose output during: ☑ compilation ☑ upload

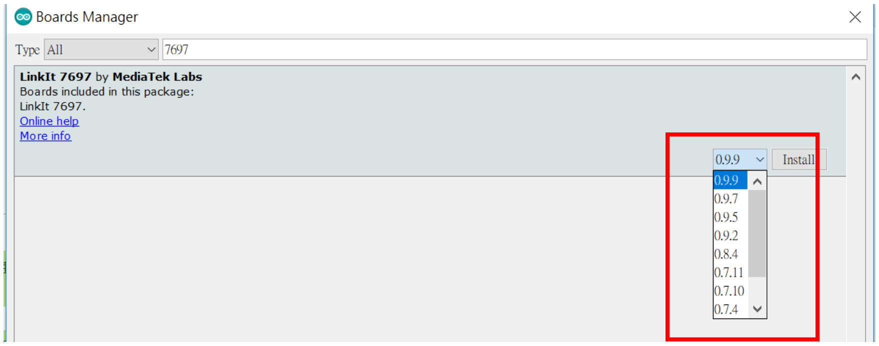
Compiler warnings: None

☑ Display line numbers
☐ Enable Code Folding
☑ Verify code after upload
☐ Use external editor
☑ Check for updates on startup
☑ Update sketch files to new extension on save (.pde -> .ino)
☑ Save when verifying or uploading

Additional Boards Manager URLs: http://download.labs.mediatek.com/package_mtk_linkit_7697_index.json

More preferences can be edited directly in the file
C:\Users\user\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

3. Open the **Tools/ Board/ Board Manager**, then search for "**7697**" and install the latest version of 7697 SDK.



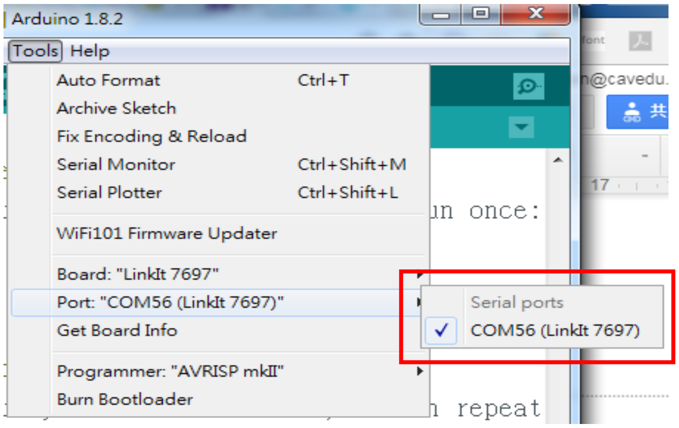
Type All 7697

LinkIt 7697 by **MediaTek Labs**
Boards included in this package:
LinkIt 7697.
Online help
More info

0.9.9
0.9.9
0.9.7
0.9.5
0.9.2
0.8.4
0.7.11
0.7.10
0.7.4

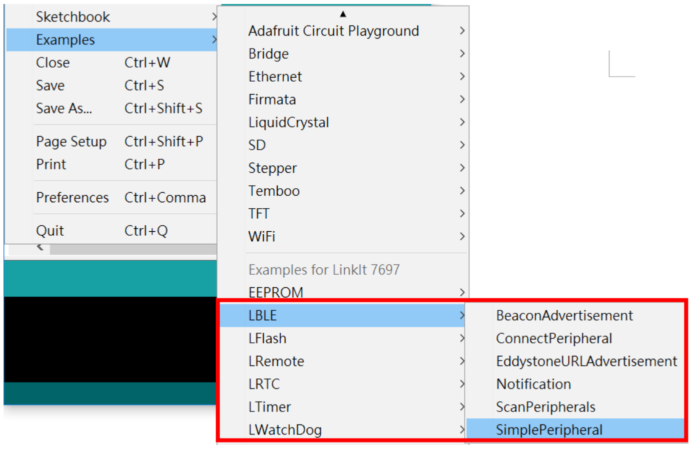
4. Download and install **CP2102N driver**(Windows / MAC/OSX) , then check the COM port in your Device manager. Look for "**Silicon Labs CP210 USB to UART Bridge(COMXX)**", this is the COM port number of your LinkIt 7697.

5. Finally, go back to Arduino IDE and check to see if IDE has recognized your LinkIt 7697 successfully from the **Tools/Port** menu. For MAC users, it should be something like "**/dev/tty.usbserialXXX**…" and forWindows users, please check the picture below:



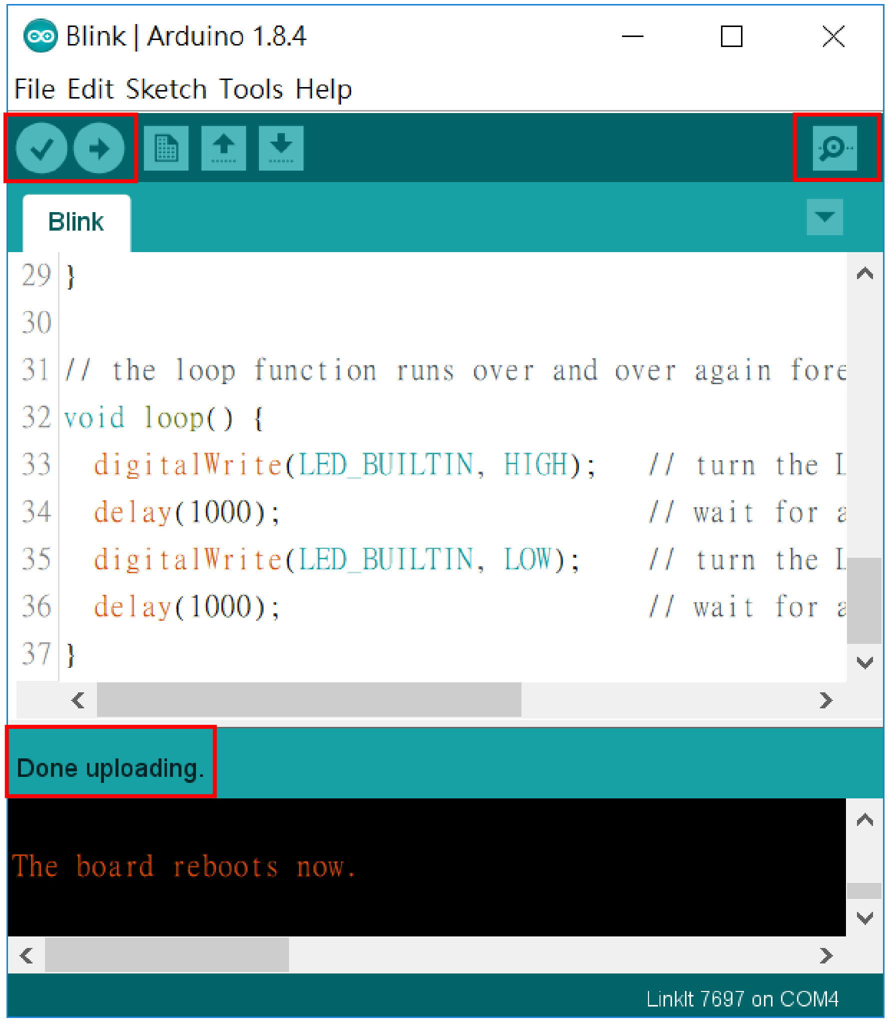
Run the Arduino sketch of this project

1. For safety reasons, not every board is marked with its Bluetooth address on the board (Arduino 101 is an exception). In Arduino IDE, first set the board to "**LinkIt 7697**" then open example "**SimplePeripheral**" from **File/Examples/LBLE** menu.



2. Set the Board from **Tools/ Board** menu to "**LinkIt 7697**" and set the COM port from **Tools/ COM port** to what you’ve discovered in Device Manager.

3. Connect the LinkIt 7697 to your computer with a micro-USB cable.
4. Press the "**Upload**" right-arrow button, this will compile and upload the Arduino sketch in Arduino IDE to your LinkIt 7697. Please make sure you see the "**done uploading**" message in the console below before you continue.



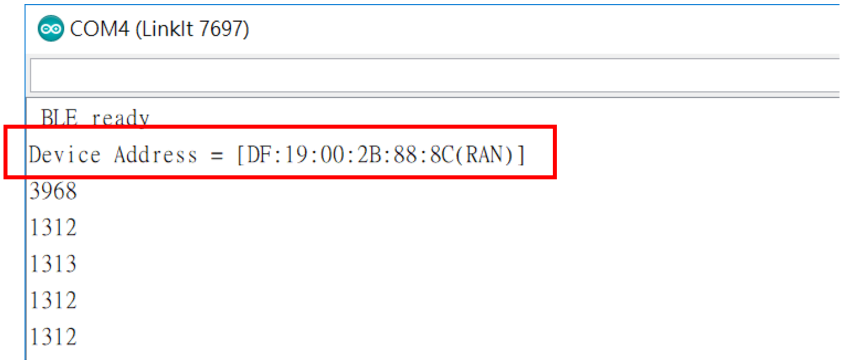
```
}

// the loop function runs over and over again fore
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the L
  delay(1000);                       // wait for a
  digitalWrite(LED_BUILTIN, LOW);    // turn the L
  delay(1000);                       // wait for a
}
```

Done uploading.

```
The board reboots now.
```

LinkIt 7697 on COM4

5. Click the magnifier icon in the up-right corner of Arduino IDE. You should see a message in the pop-up windows. The [**XX:XX:XX:XX:XX:XX**] 12-digit string is the Bluetooth address of your LinkIt 7697 and the following numbers are the moisture sensors data. We have to modify the **addr** variable value of your AI2 project. Now your LinkIt 7697 is ready to send sensor data to App Inventor.



```
 BLE ready
Device Address = [DF:19:00:2B:88:8C(RAN)]
3968
1312
1313
1312
1312
```

App Inventor

The purpose of this project is to interact with the LinkIt 7697 dev board with App Inventor through BLE communication. The main idea is to send moisture sensor data, which is connected to the LinkIt 7697's analog pin #A0, to App Inventor, and change images according to the pin status.

Now login to your App Inventor account and create a new project.

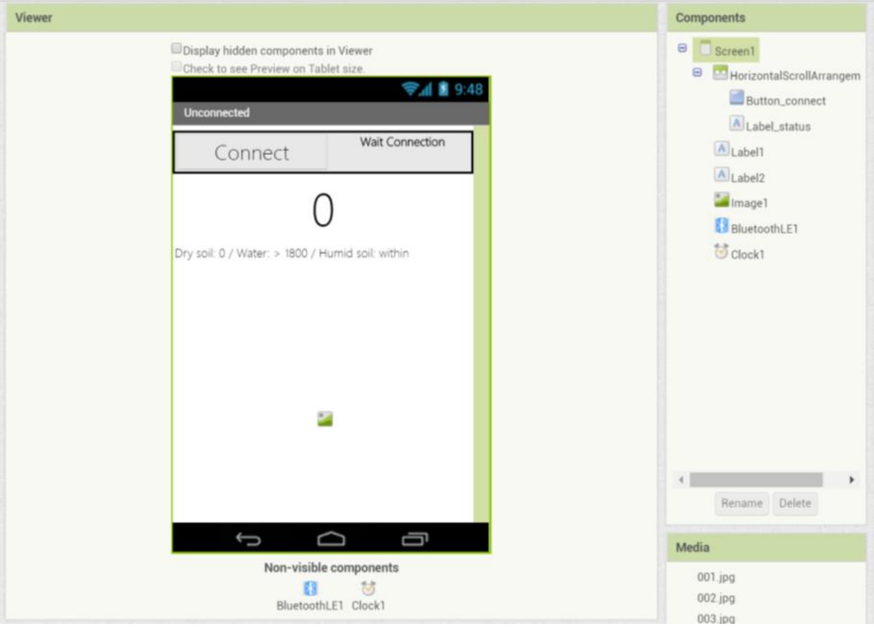
Designer

1. First, import the BLE extension from this URL:
   - http://iot.appinventor.mit.edu/assets/com.bbc.micro:bit.profile.aix
   - and add it to the project by dragging it into Viewer.
2. Add one **HorizontalArrangement** component from Layout drawer and set its width to "**Fill Parent…**"
3. Add one **Button** and one **Label,** then put them into a **HorizontalArrangement** component. The Button is used to connect to and disconnect from your Bluetooth device. The Label is used to show connection status.
4. Add two more **Label**s, one for displaying data from LinkIt 7697, the other for messages (Modify its text to "**Dry soil: 0 / Water: > 1800 / Humid soil: within**").
5. Add a **Clock** from the **Sensor** drawer, and set its **TimerInterval** to **100** (milliseconds). We use it to read the LinkIt 7697's

moisture sensor data 10 times per second.

6. Add an **Image** from **Media** drawer.
7. Upload three images you would like to use to represent three states of your plant.

After some adjusting, your design should be something like this. Feel free to modify the component's background color, position and text size.

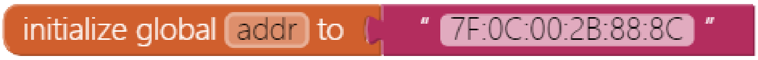


Blocks

Let's look at our blocks step by step:

1. **Variable for Bluetooth address**

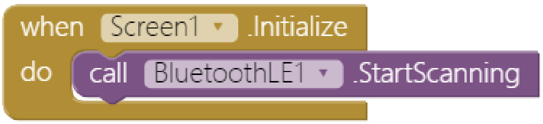
   Please replace the value with what you get from Arduino's Serial Monitor.



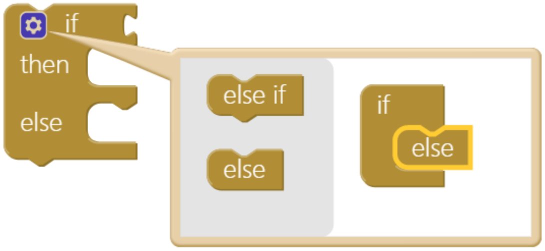
```
initialize global addr to " 7F:0C:00:2B:88:8C "
```

2. **Initialize and connect**

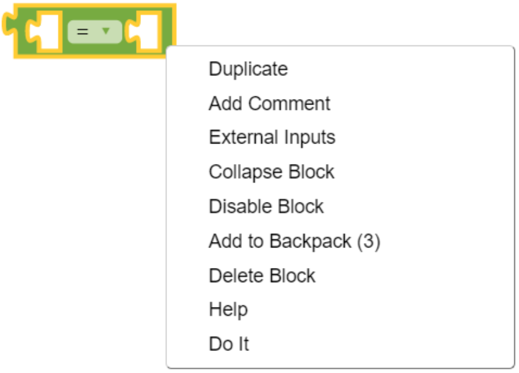
In **Screen1.Initialize** event, we ask **BluetoothLE** component to scan for BLE devices nearby(**BluetoothLE1.StartScanning**).



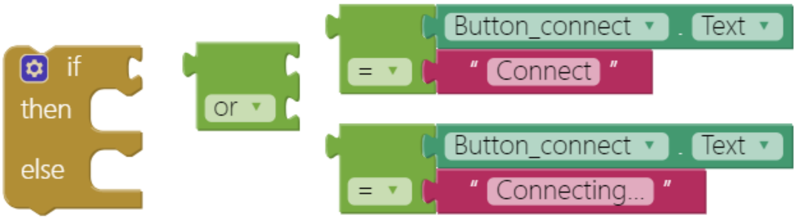
3. **Button to connect and disconnect from the Bluetooth device**
   In **Button_connect.Click** event, we are going to connect or disconnect from Bluetooth device depending on the button text. First, add an **if** condition, then click its blue gear to add an **else**.



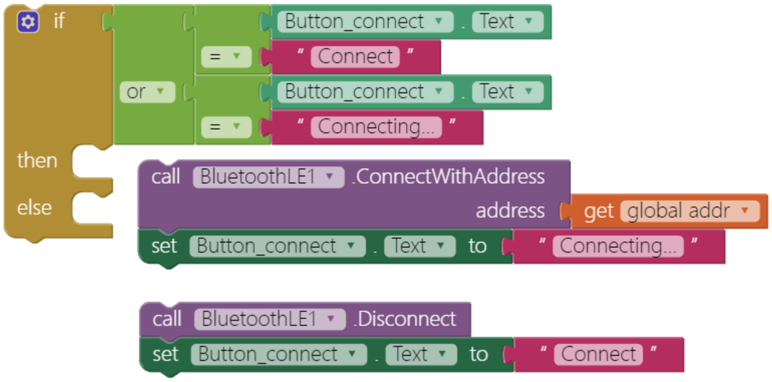
Add an **or** command from logic block, then right-click click it and select "**External Inputs**". this will make it into more rectangular shape with input on the right hand side.



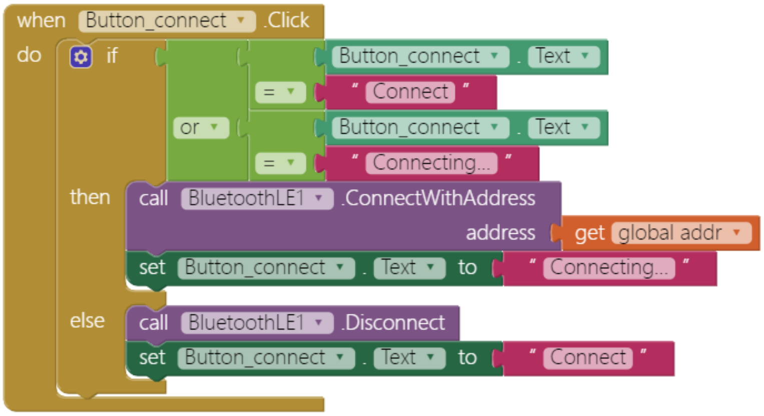
Now we want to check whether **Button_connect.Text** is "**Connect**" OR "**Connecting...**", this is how App Inventor decide to connect or disconnect Bluetooth connection with LinkIt 7697. please combine these blocks.



Good! When the **Button_connect**’s text is "**Connect**" OR "**Connecting...**", we will connect to the specified Bluetooth device (**BluetoothLE1.ConnectwithAddress**), which is our LinkIt 7697. If not, then disconnect (**BluetoothLE1.Disconnect**). In both cases, we will show a message on **Button_connect**.

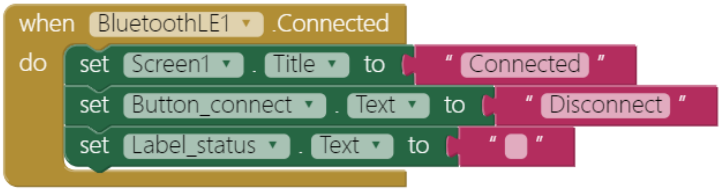

if
or
= Button_connect . Text
" Connect "
= Button_connect . Text
" Connecting... "
then
call BluetoothLE1 .ConnectWithAddress
address get global addr
set Button_connect . Text to " Connecting... "
else
call BluetoothLE1 .Disconnect
set Button_connect . Text to " Connect "

Put everything into one large **Button_connect.Click** event, the final product should look like this:


when Button_connect .Click
do if
or
= Button_connect . Text
" Connect "
= Button_connect . Text
" Connecting... "
then call BluetoothLE1 .ConnectWithAddress
address get global addr
set Button_connect . Text to " Connecting... "
else call BluetoothLE1 .Disconnect
set Button_connect . Text to " Connect "

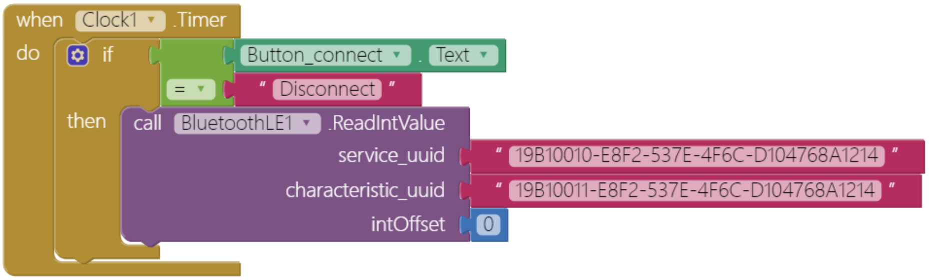
4. BLE Connected

When connected successfully (**BluetoothLE.Connected** event), we show related messages on several components.



```
when BluetoothLE1 .Connected
do  set Screen1 . Title to " Connected "
    set Button_connect . Text to " Disconnect "
    set Label_status . Text to " "
```

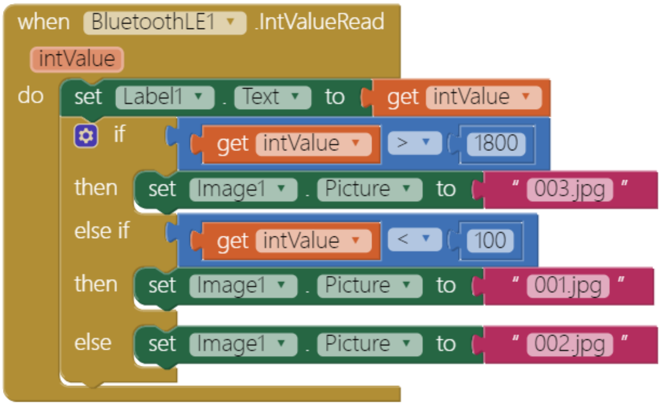
5. **Ask for LinkIt 7697's moisture sensor data periodically**
   Since we would like to check the moisture status periodically, it is best to use a Clock component with its timer event.
   In the **Clock.Timer** event, we first check the connection and then request data from LinkIt 7697 using the **BluetoothLE.ReadInt** method. Notice that the **service_uuid** and **characteristic_uuid** must be identical with what is in the Arduino sketch.



```
when Clock1 .Timer
do  if    Button_connect . Text = " Disconnect "
    then  call BluetoothLE1 .ReadIntValue
                   service_uuid " 19B10010-E8F2-537E-4F6C-D104768A1214 "
            characteristic_uuid " 19B10011-E8F2-537E-4F6C-D104768A1214 "
                      intOffset 0
```

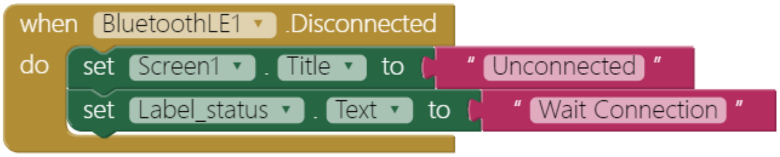
6. **Show received data on the label, and change the image**
   If it is read successfully, the **BluetoothLE.IntValueRead** event will be triggered. In this event, we first show the received data on Label1, and change **Image1** component's Picture to match. If the value is greater than 1800, that means the soil is too wet, so change the picture to **003.jpg**. If the value is smaller than 100, that means too dry, so change the picture to **001.jpg**. If it is within the 100 to 1800 range, that means the soil is well moisturized, so change the picture to **002.jpg**. Check the last section for the screenshots.

Note: the analog pin data ranges from 0 to 4095 (12-bit resolution). This is a bit different from Arduino (0 to 1023, 10-bit resolution).



```
when BluetoothLE1 .IntValueRead
  intValue
do  set Label1 . Text to  get intValue
    if    get intValue  >  1800
    then  set Image1 . Picture to  " 003.jpg "
    else if  get intValue  <  100
    then  set Image1 . Picture to  " 001.jpg "
    else  set Image1 . Picture to  " 002.jpg "
```

7. **Disconnect**
   Disconnect by clicking the **Button_connect** or pressing the USS button(D6) of LinkIt 7697. This will reset the app to its initial state, to wait for the next connection request.



```
when BluetoothLE1 .Disconnected
do  set Screen1 . Title to  " Unconnected "
    set Label_status . Text to  " Wait Connection "
```

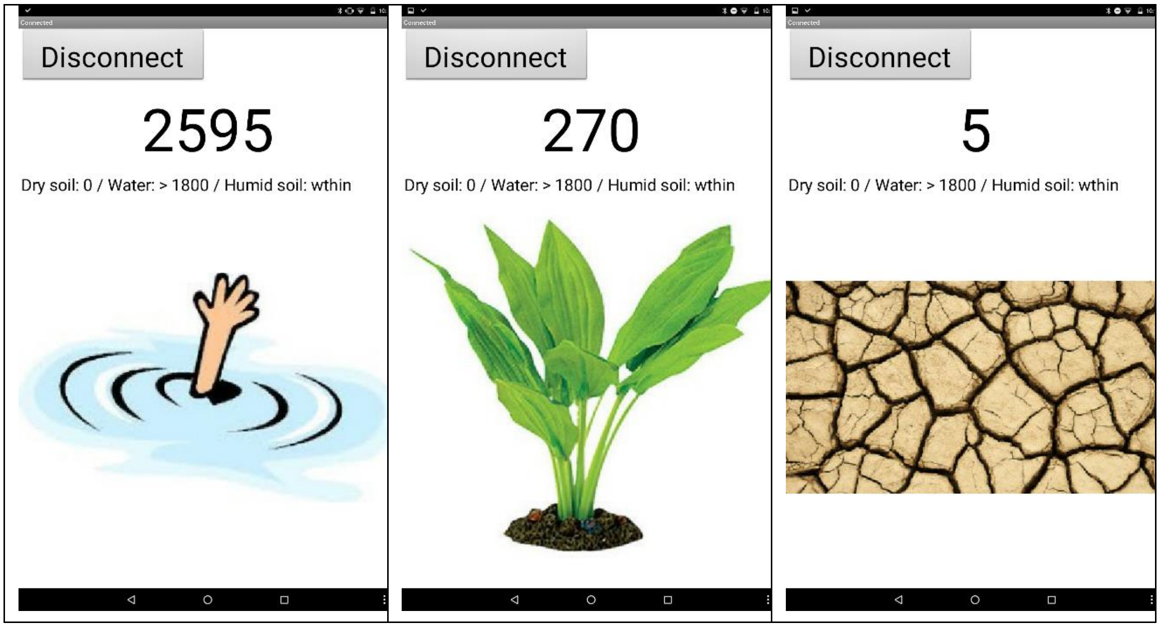
Arduino code

Please download from here and upload to your LinkIt 7697.

Tips

Make sure your LinkIt 7697 is running correctly as a BLE peripheral. Open your app and click **Connect** button, you should see the larger number in the middle of screen is varying according to your moisture sensor.

Get a cup of water, put the moisture sensor into the water, and see how the value and picture change!



2595

Dry soil: 0 / Water: > 1800 / Humid soil: wthin

270

Dry soil: 0 / Water: > 1800 / Humid soil: wthin

5

Dry soil: 0 / Water: > 1800 / Humid soil: wthin

Brainstorming

1. Try to use other analog input components like light sensor or potentiometer.
2. Try to add more images and show them when the sensor data is within a specific range.