Bluetooth Based IOT through Arduino

G V V Sharma*

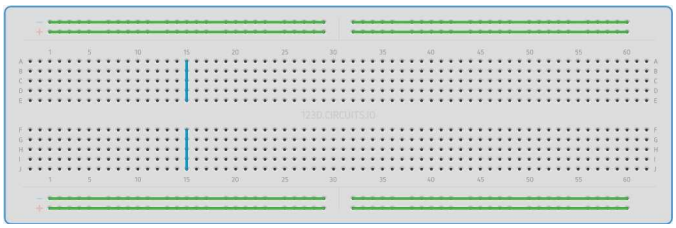


Fig. 1: Breadboard

1 Measuring the resistance

Problem 1. Connect the 5V pin of the Arduino to an extreme pin of the Breadboard shown in Fig. 1. Let this pin be V_{cc}.

Problem 2. Connect the GND pin of the Arduino to the opposite extreme pin of the Breadboard.

Problem 3. Let R_1 be the known resistor and R_2 be the unknown resistor. Connect R_1 and R_2 in series such that R_1 is connected to GND and R_2 is connected to V_{cc}. Refer to Fig. 3

Problem 4. Connect the junction between the two resistors to the A0 pin on the Arduino.

Problem 5. Connect the arduino to the computer so that it is powered.

Problem 6. Open the Arduino IDE and type the following code. Open the *serial monitor* to view the output.

```
//Declarations
int V_out_q=0;
//V_out_q is the quantized voltage
float V_in = 5,V_out;
//V_in = V_cc
float R1=220,R2;
//R1 is known resistance
//R2 is unknown resistance
```

*The author is with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

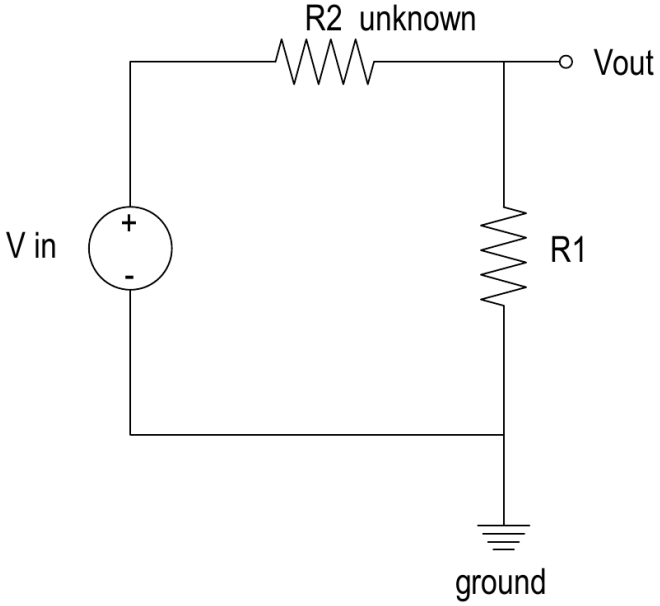


Fig. 3: Voltage Divider

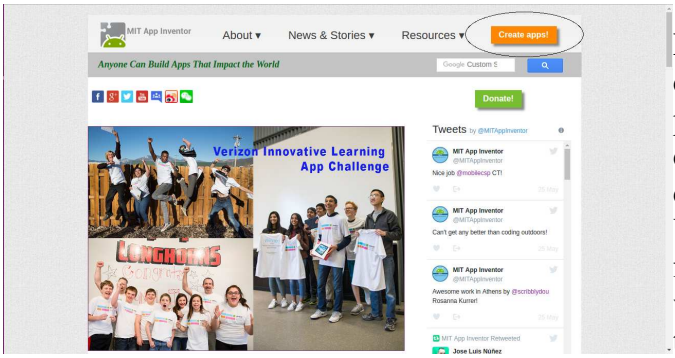
```
void setup()
{
//Get the result onto the serial
   monitor
  Serial.begin(9600);
}

void loop()
{
//V_0ut_q is an integer between 0
   to 1023
V_out_q=analogRead(0);//reading
   from A0

//V_out is the actual voltage at
   the junction of R1 and R2
V_out = V_in*V_out_q/1024;

R2 = R1*((V_in)/(V_out)-1.0);
delay(3000);
Serial.println(R2);
}
```

2 Development of Android Application to Display the Measured Resistance via Bluetooth

In the following, we make an android application using MIT App Inventor or Android studio (here we are using MIT App Inventor) to display the measured resistance.

Problem 7. Click on link http://appinventor.mit.edu/ for using MIT App Inventor.

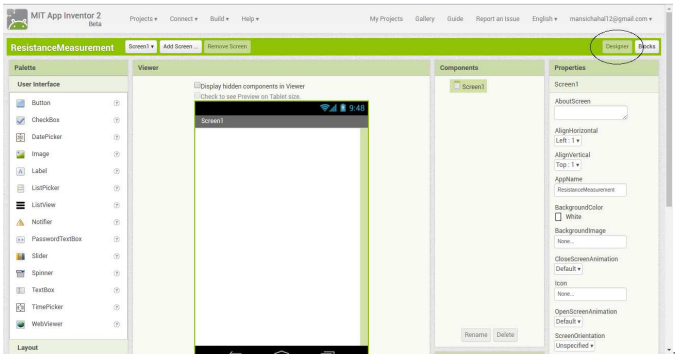
Problem 8. Click on Create apps! tag to get started.



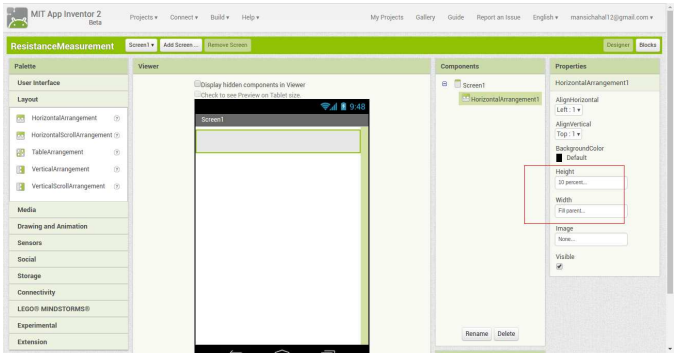
Problem 9. Log in to App Inventor with a gmail user name and password.

Problem 10. Start a new project. Type in the project name (underscores are allowed, spaces are not) and click OK

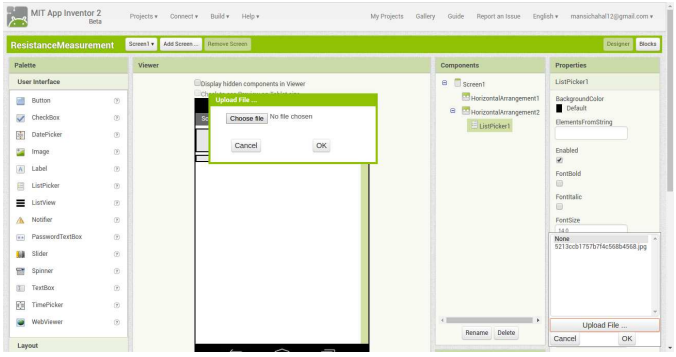
Problem 11. You are now in the Designer editor, where you lay out the ”user interface” of your app



Problem 12. Click on ”Layout” in the palette section. Then, click and hold on ”HorizontalArrangement” and drag your mouse over to the Viewer. Drop the HorizontalArrangement and a new HorizontalArrangement will appear on the Viewer. This HorizontalArrangement is just to leave some blank space before placing something else for making the layout effective. Go to the properties of this HorizontalArrangement. Change the Height property to 10 percent and Width to ”Fill parent”. You can change the properties according to your requirement



Problem 13. Drag another HorizontalArrangement component onto the Viewer and change the Width property to “Fill parent”. From the User Interface component group, select and drag the “ListPicker” component onto the 2nd HorizontalArrangement on Viewer. The ListPicker component provide similar function as Button that function like menu options, where you can click on one of the selection to get to the specified option. Remove the text from the Text property of ListPicker and change the AlignHorizontal property of HorizontalArrangement to “Center”. Download a .PNG bluetooth icon from the net. Upload this image in the Image property of ListPicker for the symbol of bluetooth. After uploading, change the Width and Height property of the ListPicker to 70 pixels. You won’t see any bluetooth icon now. Wait.



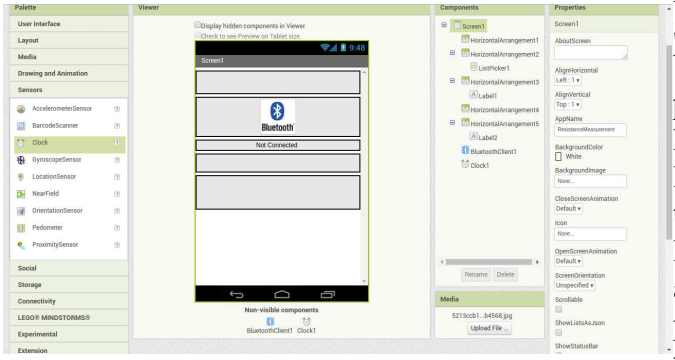
Problem 14. Drag another HorizontalArrangement component onto the Viewer and change the Width property to “Fill parent”. From the User Interface component group, select and drag the “Label” component onto the 3rd HorizontalArrangement on Viewer. Change the Text property of Label to Not Connected. Change the AlignHorizontal property of HorizontalArrangement to “Center”.

Problem 15. Again drag a HorizontalArrangement component onto the Viewer. Change the Width property to “Fill parent” and Height to 8 percent. This 4th HorizontalArrangement is also to leave some blank space before placing something else.

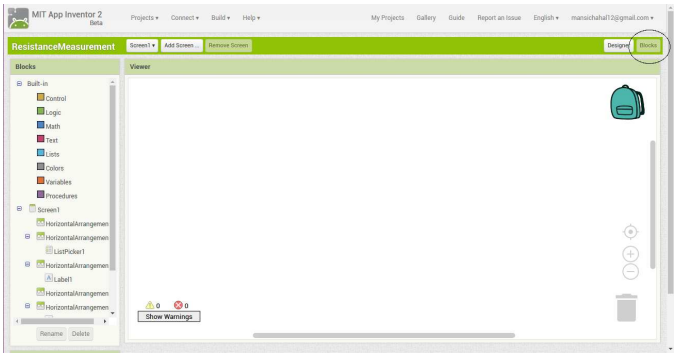
Problem 16. Drag 5th HorizontalArrangement component onto the Viewer.Change the Height property to 15 percent and Width to ”Fill parent”. From the User Interface component group, select and drag the 2nd ”Label” component onto the 5th HorizontalArrangement on Viewer. Change the AlignHorizontal and AlignVertical property of HorizontalArrangement to ”Center”. Remove the text from the Text property of Label and change the Width property to ”Fill parent”. Change the Text alignment property of Label to ”Center” and Height to 5 percent.

Problem 17. From the Connectivity component group in the Palette section, select and drag the BluetoothClient component to the Viewer. Since the BluetoothClient component does not have user interface, it is a non-visible component and not shown on the Viewer.

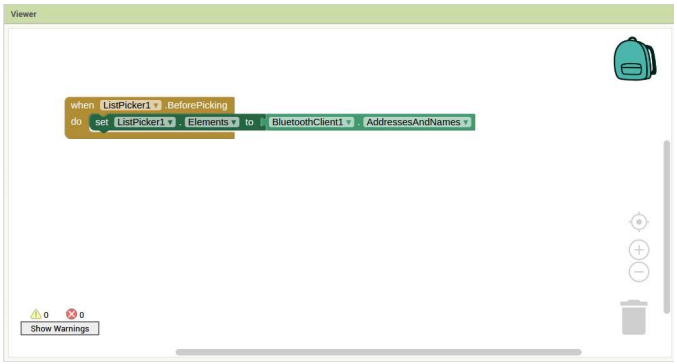
Problem 18. Next, from the Sensors component group in the Palette section, select and drag the Clock component onto the Viewer. The Clock component is a non-visible component and does not shows up on the app screen. This is the user interface layout of the app.



Problem 19. Next, with all of the required layout components in place, click on the Blocks button to switch to the Block editor to add programming logic.



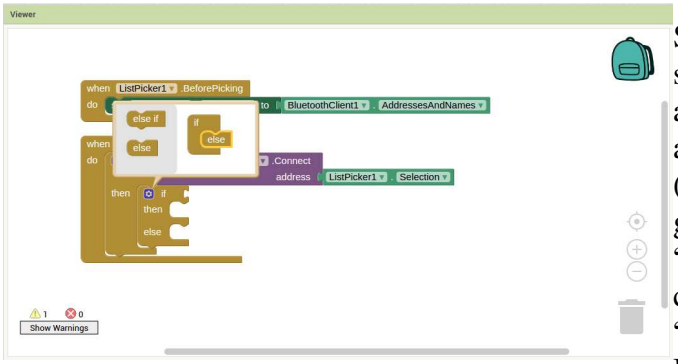
Problem 20. From the Screen1\ListPicker1 component group, select and add the “when ListPicker1.BeforePicking” and “set ListPicker1.Elements” components. From the Screen1\BluetooothClient1 component group, select and add the “BluetoothClient1.AddressAndNames” component to the Viewer section. Assemble the blocks to get the following image.



The above group of components provide the function to retrieve the list Bluetooth devices paired with the Android device.

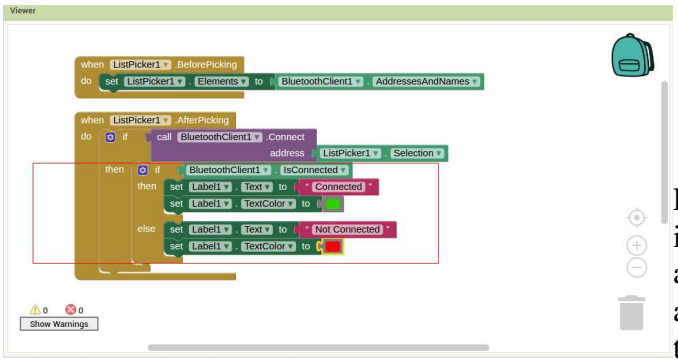
Problem 21. Continue and add the “when ListPicker1.AfterPicking” component from the Screen1\ListPicker1 component group to the Viewer section. The ListPicker1.AfterPicking component is an event handler after an item is selected. From the Built-in \Control group, select and add the “If then” component, a conditional handler, to the Viewer section. Next, select and add the “call BluetoothClient1.Connect address” (from the Screen1\BluetooothClient1 component group). and “ListPicker1.Selection” (from the Screen1\ListPicker1 component group) components and link to the “if” condition. Again, from the Built-in \Control group, select and add the “If then” component and link to the “then” condition. To extend the block with as many else and else if branches click the blue icon. Do this to get the

following image.



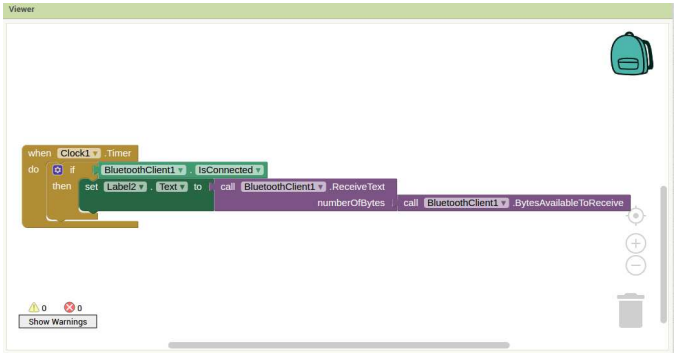
Problem 22. Next, select and add the "BluetoothClient1.IsConnected" (from the Screen1\BluetooothClient1 component group) and link to the second "if" condition. Select "set Label1.Text" (from the Screen1\Label1 component group) components and link to the "then" condition and from the Built-in \Text, click on the blank text entry component and enter the word Connected. Select "set Label1.TextColor" (from the Screen1\Label1 component group) components and link just below the "Label1.Text" and from the Built-in \Colors, click on the green color.

Problem 23. Again, select "set Label1.Text" (from the Screen1\Label1 component group) components and link to the "else" condition and from the Built-in \Text, click on the blank text entry component and enter the word Not Connected. Select "set Label1.TextColor" (from the Screen1\Label1 component group) components and link just below the "set Label1.Text" and from the Built-in \Colors, click on the red color. Make sure that you get the following image.



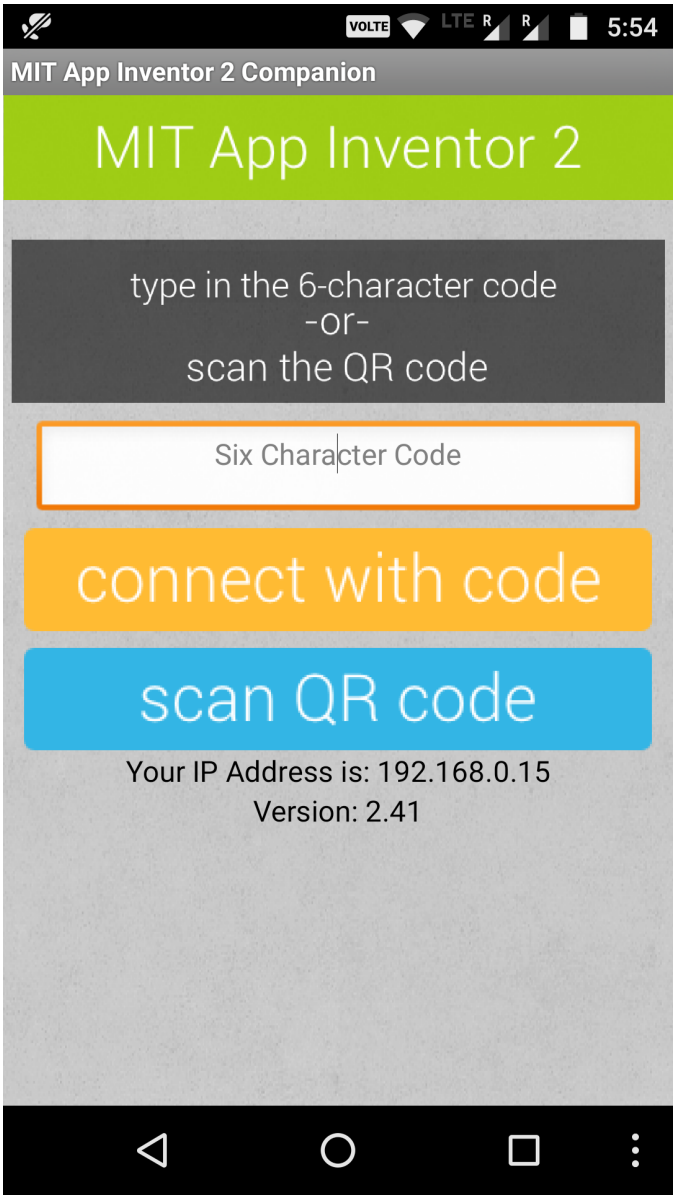
The group of components, highlighted within the red rectangular frame, is part of an event handler to change the status that either Bluetooth device is connected or not.

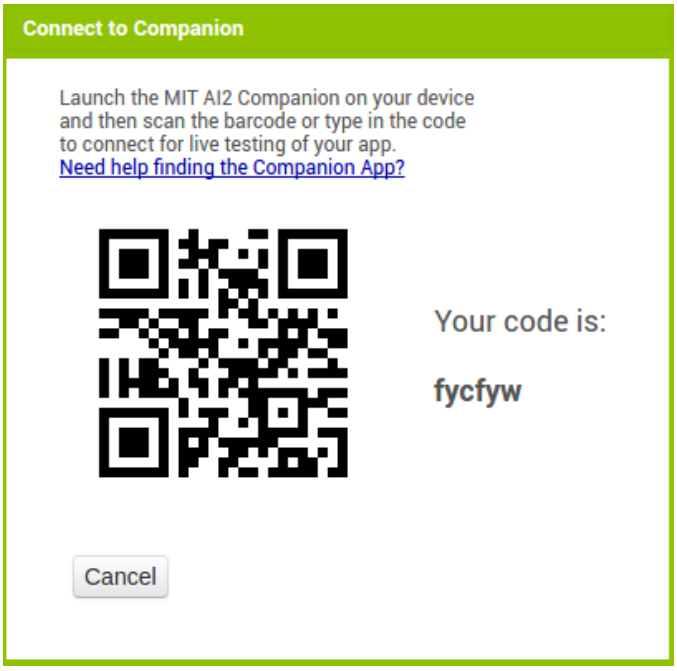
Problem 24. Continue and add the "when Clock1.Timer" component from the Screen1\Clock1 component group to the Viewer section. From the Built-in \Control group, select and add the "If then" component. Next, select and add the "BluetoothClient1.IsConnected" (from the Screen1\BluetooothClient1 component group) and link to the "if" condition and select "set Label2.Text" (from the Screen1\Label2 component group) component and link to the "then" condition. Select and add the "call BluetoothClient1.ReceiveTextnumberOfBytes" and "call BluetoothClient1.BytesAvailabelToReceive" (from the Screen1\BluetooothClient1 component group) components. Link them to get the following image.



The above group of function block provide the program logic to display the measured resistance value, via the BluetoothClient1 connection to the connected Bluetooth device.

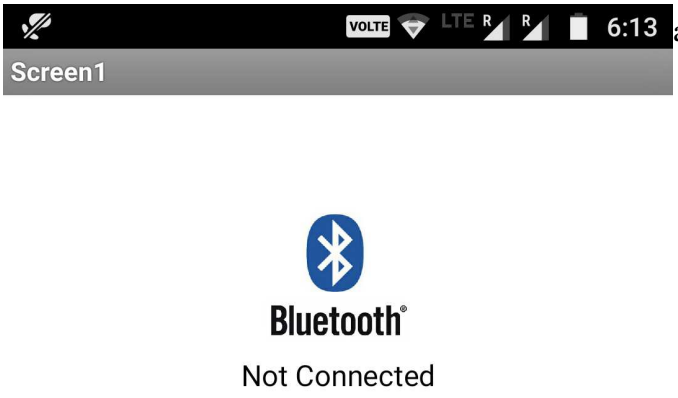
Problem 25. At this point, we have all of the intended function for the app. Before testing the app, we need to establish connectivity to a device or an emulator. To use a real device, you need to install the MIT AI2 Companion app from the App Store. Install and launch the app on the target device. After the MIT AI2 Companion app is launched, you have the option to enter a six digit code or use the scan QR code option to connect to App Inventor



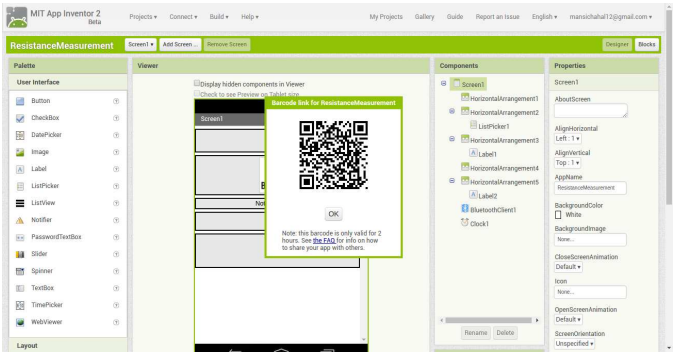


Problem 26. From the App Inventor's Connect menu, click on AI Companion to bring up the following screen and wait. Make sure that your computer and mobile device are connected to the same WiFi network.

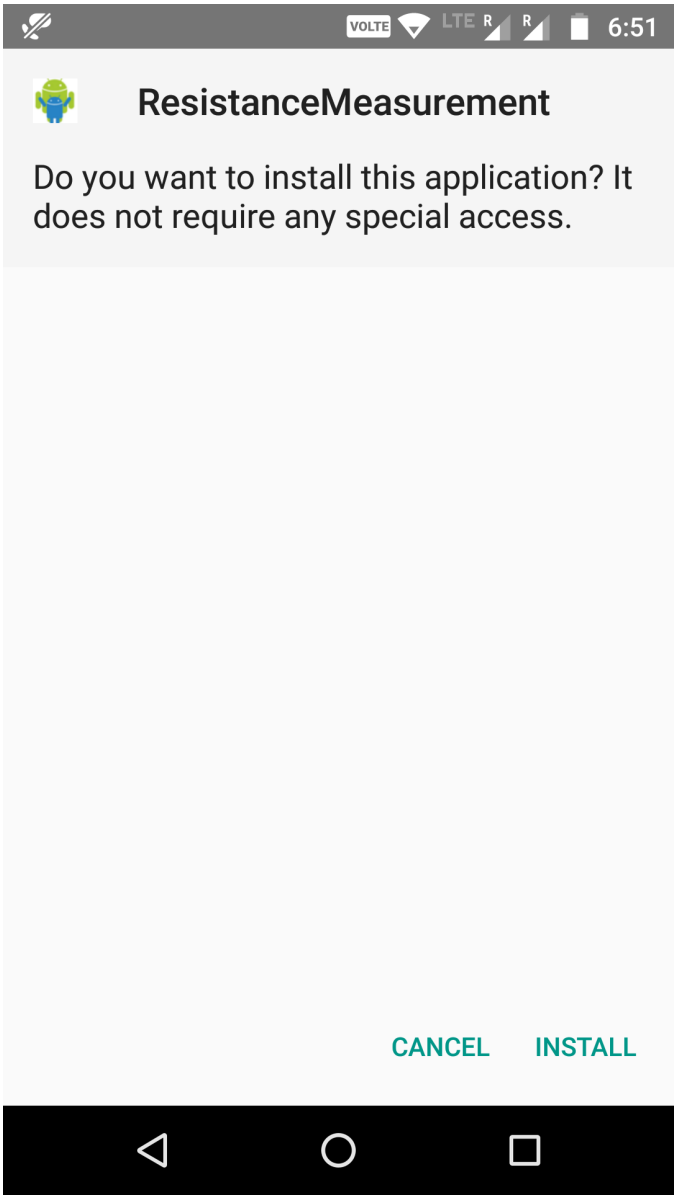
Problem 27. From the target device, you can enter the six digit code, or scan the QR code to establish connectivity to the App Inventor. Once connected, the app will display on the device, as shown below. Note that this does not install the app on your device.

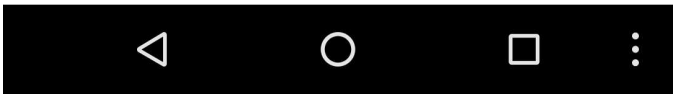


app, as shown below

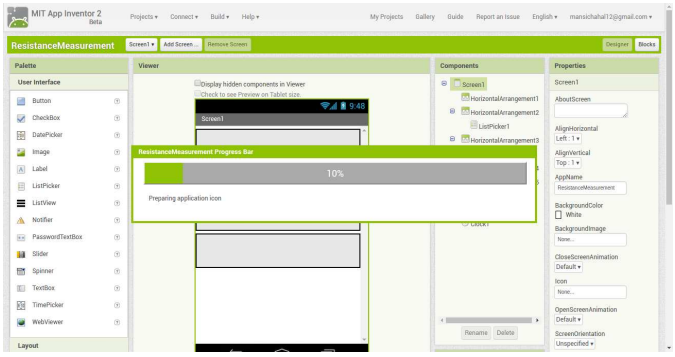


Problem 30. After scanning the QR code using MIT AI2 Companion, the following screen is shown on the device, asking for permission to install the app





Problem 28. From the App Inventor Build menu, click on App(provide QR code for .apk) to build the app. App Inventor shows the following progress as it builds the app



Problem 29. After the build is done, a QR code is provided for the MIT AI2 Companion to install the Now install and launch the app on the device.

3 Displaying Measured Resistance on Android App via Bluetooth

Power off the Arduino.

Problem 31. Connect the TX pin of the Arduino to R_2.

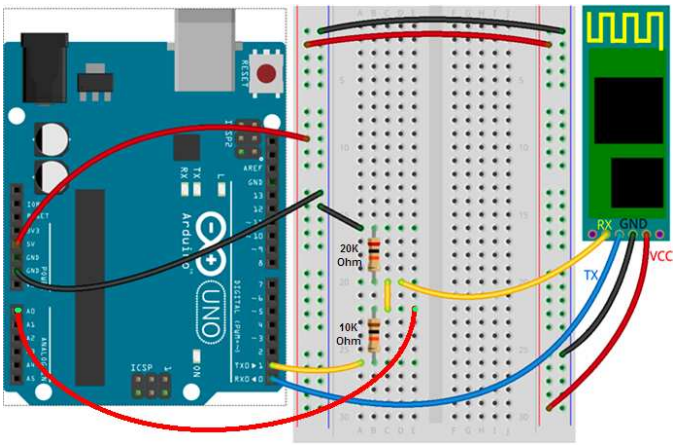
Problem 32. Connect the other end of R_2 to R_1

Problem 33. Connect the other end of R_1 to GND.

Problem 34. Connect A0 to RX pin of the Bluetooth module.

Problem 35. Connect TX of Bluetooth to RX of Arduino.

Problem 36. Make the V_cc and GND connections for the Bluetooth module. The final connection digram is available below.



Problem 37. Power up the Arduino and get the code in Section I running. Connect to the the bluetooth module and you should be able to see the resistance value on the app developed using the app inventor.

4 Explanation

1) We create a variable called analogPin and assign it to 0. This is because the voltage value we are going to read is connected to analogPin A0.
2) The 10-bit ADC can differentiate 1024 discrete voltage levels, 5 volt is applied to 2 resistors and the voltage sample is taken in between the resistors. The value which we get from analogPin can be between 0 and 1023. 0 would represent 0 volts falls across the unknown resistor. A value of 1023 would mean that practically all 5 volts falls across the unknown resistor.
3) V_{out} represents the divided voltage that falls across the unknown resistor.
4) The Ohm meter in this manual works on the principle of the voltage divider shown in Fig. 3.

$$V_{out} = \frac{R_1}{R_1 + R_2} V_{in} \tag{37.1}$$

$$\Rightarrow R_2 = R_1 \left(\frac{V_{in}}{V_{out}} - 1 \right) \tag{37.2}$$

In the above, $V_{in} = 5\text{V}$, $R_1 = 220\Omega$.