

Arduino and Android using MIT app inventor 2.0

For Everyone

Magesh jayakumar

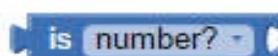
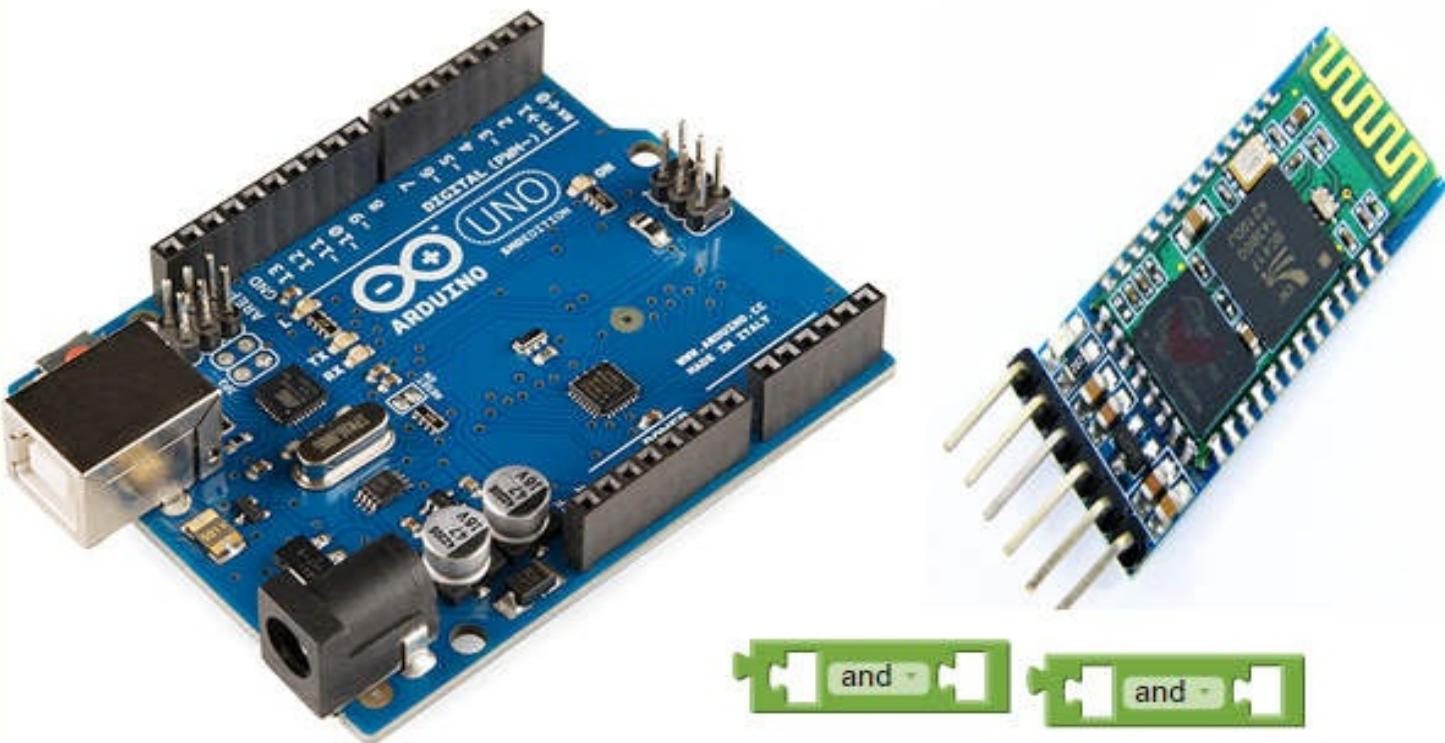


Table of Contents

Introduction

[What is MIT App Inventor?](#)

[Arduino \(From Arduino Website\)](#)

Turn On and OFF an LED (Hello World)

[Creating Android App](#)

[Connecting Bluetooth and programming Arduino](#)

Control LED by Speech recognition

[Task](#)

Control a Servo Motor

[Arduino code for servo control](#)

Control Multiple Servo Motors

[Arduino Code Multiple servo](#)

RGB Color Slider

[Activity starter](#)

[Complete program for the RGB Slider](#)

[Arduino Program for RGB Slider](#)

[Common anode Arduino program](#)

[Common Cathode Arduino program](#)

RGB Color Picker

[Arduino Program](#)

Temperature data into Android

[Arduino Program for Temperature](#)

Android Remote control Robot

[Arduino Program for Android remote](#)

[Control the same robot by speech recognition](#)

How to import aia files to MIT app inventor

[Contact and app inventor files.](#)

Introduction

What is MIT App Inventor?

MIT App Inventor is an innovative beginner's introduction to programming and app creation that transforms the complex language of text-based coding into visual, drag-and-drop building blocks. The simple graphical interface grants even an inexperienced novice the ability to create a basic, fully functional app within an hour or less.

[Arduino \(From Arduino Website\)](#)

Arduino is an open-source prototyping platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

Turn On and OFF an LED (Hello World)

Before starting to go inside this chapter, firstly you need to install Arduino Ide from the website arduino.cc

Download the latest Arduino software and install it on your laptop or PC, without Arduino IDE you cannot program your Arduino board.

Things you need for this tutorial are 1) Arduino board 2) LED and Android phone to control the LED connected to Arduino.

First we create android app to control the LED, to do this you should have a Gmail account or you can create a new account it's really easy and most of people will have an existing Gmail account.

The following step-by-step picture tutorial will guide you to create your first android app that connects to your Arduino.

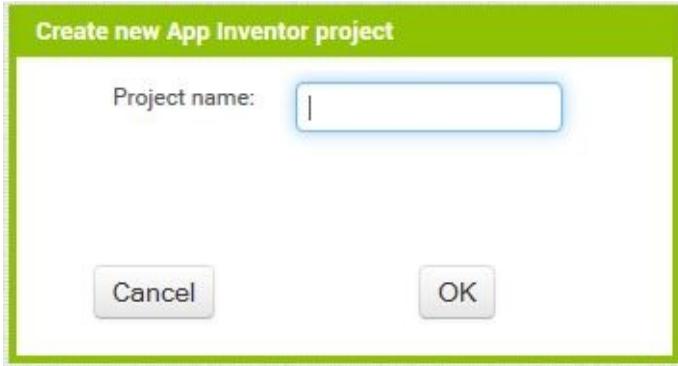
- 1) Go to the website <http://ai2.appinventor.mit.edu> , when you enter for the first time you will be required to login with your Gmail id and password.
- 2) MIT appinventor will ask to use your Gmail account click allow and check remember this approval for the next 30 days.

The screenshot shows a Google Accounts dialog box. It asks for permission to access a Google Account for the MIT AppInventor Version 2 application. The user has selected their email account (jkpol64@gmail.com). Below the account selection, there is a note that Google is not affiliated with the app. At the bottom, there are 'Allow' and 'No thanks' buttons, and a checked checkbox for 'Remember this approval for the next 30 days'.

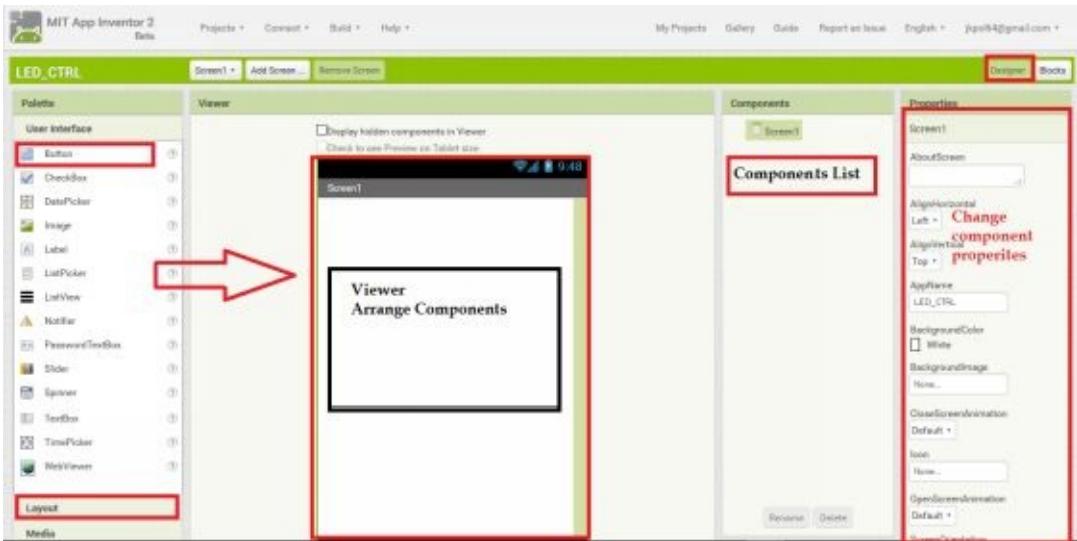
- 3) When you logged into MIT app inventor development environment for the first time, there won't be any projects, you will be greeted with an empty window. Click on the start new project to create your first android app.



4) Name your new project as LED_CTRL or whatever name you would like for your first app and click ok to enter into the main screen.



5) When you click ok after naming your app, you will be moved to a main screen to make your app, the screen that appears on your window is called as Designer, where you need to design user interface like buttons, widget, text pane, label this window act as a controller and there is also another window where you need to program the graphical codes to make your function to work.



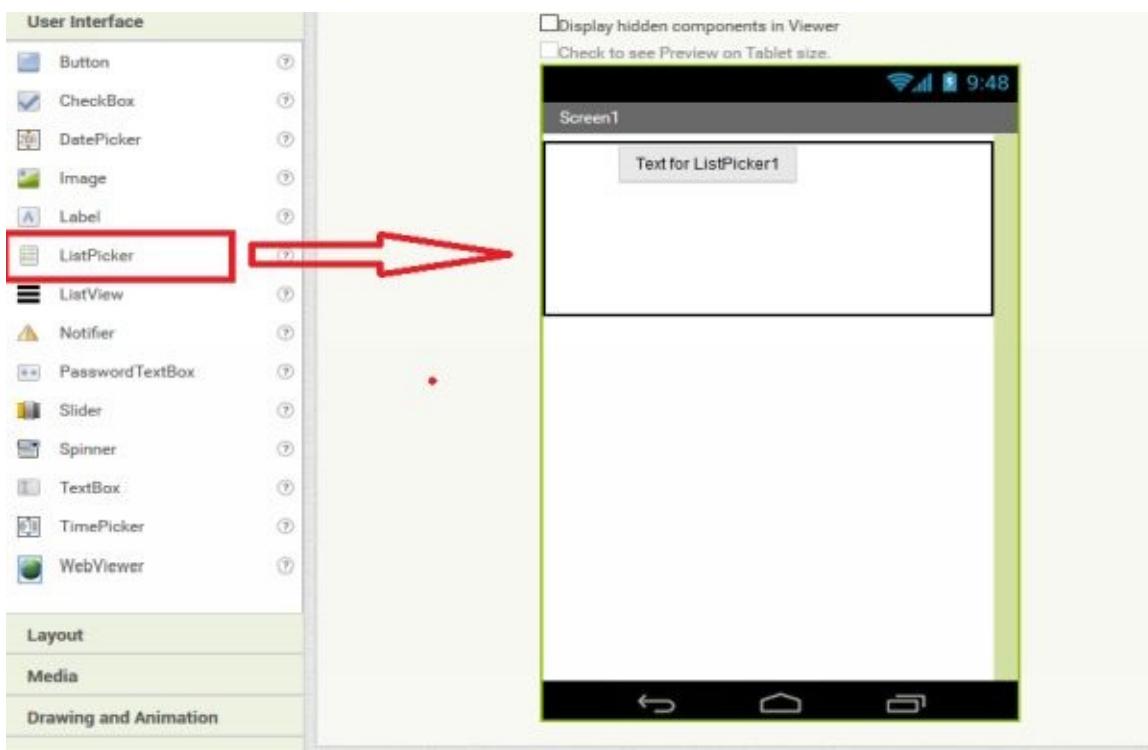
On the left side you can see palette, the palette which contains interface, layout, Media, Drawing and Animation, Sensors, Social, Connectivity etc., To create your user interface on the main screen you need to pick buttons and your required stuffs from the palette.

Click Layout on the left side below palette, Pick Table Arrangement and drop into the Viewer.

On the left side of the table properties, change columns and rows to 3 and also change the width to fill parent from automatic.



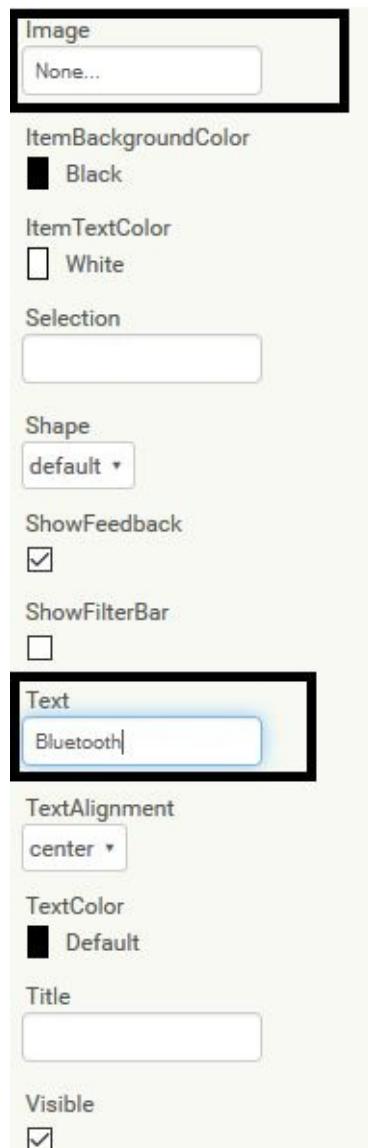
Pick Listpicker from User Interface and drop into first row in the Viewer screen. This used to connect to your Bluetooth device. When you click this button it will display available Bluetooth devices.



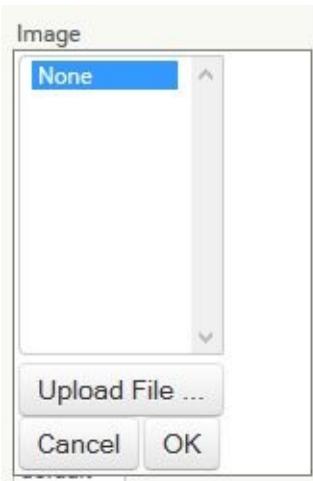
Select ListPicker from the Components List



Go to list picker properties and change the text to Bluetooth and you can also upload an image file to the list picker.



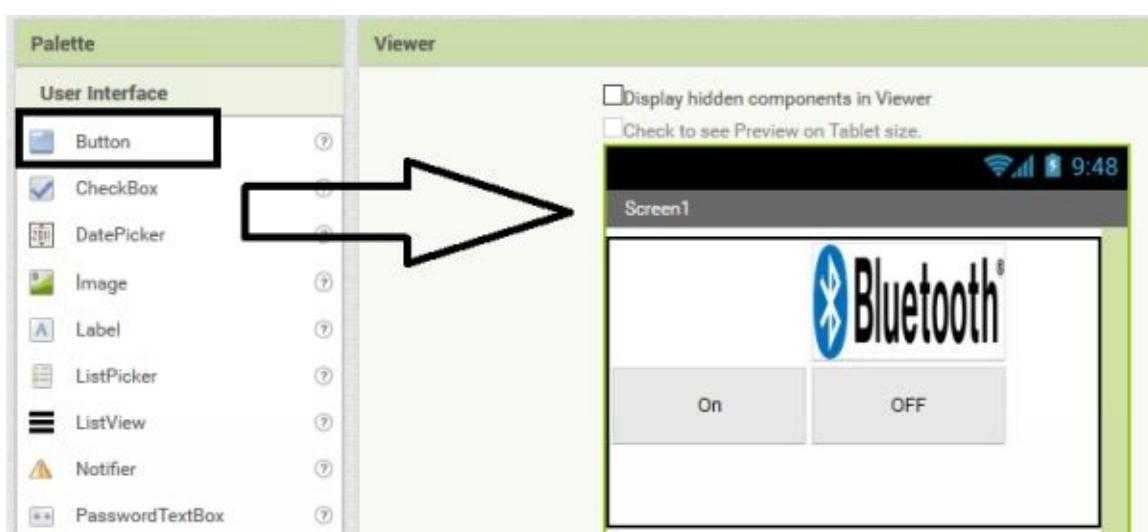
Click on the None tab below Image and upload your Bluetooth picture to set it to Listpicker.



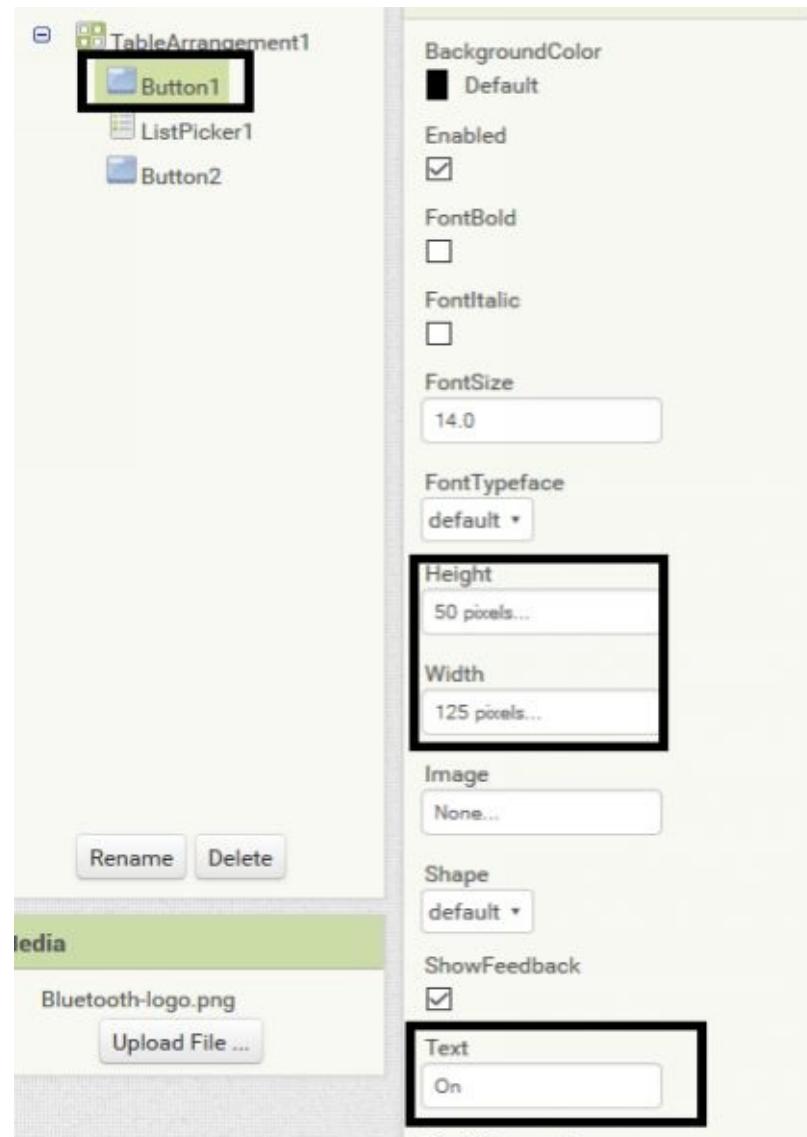
Click the upload file and browse to the image you want to use for the ListPicker, if the Viewer Screen displays the Image for the ListPicker in larger size you need to alter the height and width of the ListPicker by going to the properties for the Listpicker.



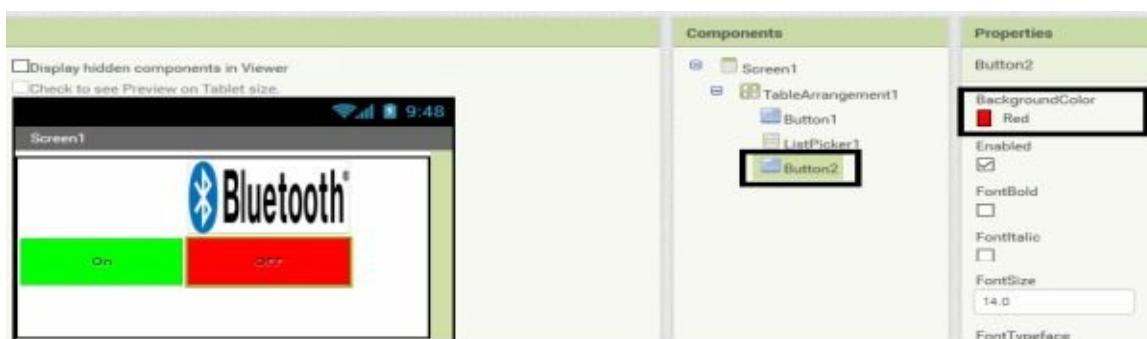
Next pick a button and drop into second row of the Viewer screen and add another button to the same row.



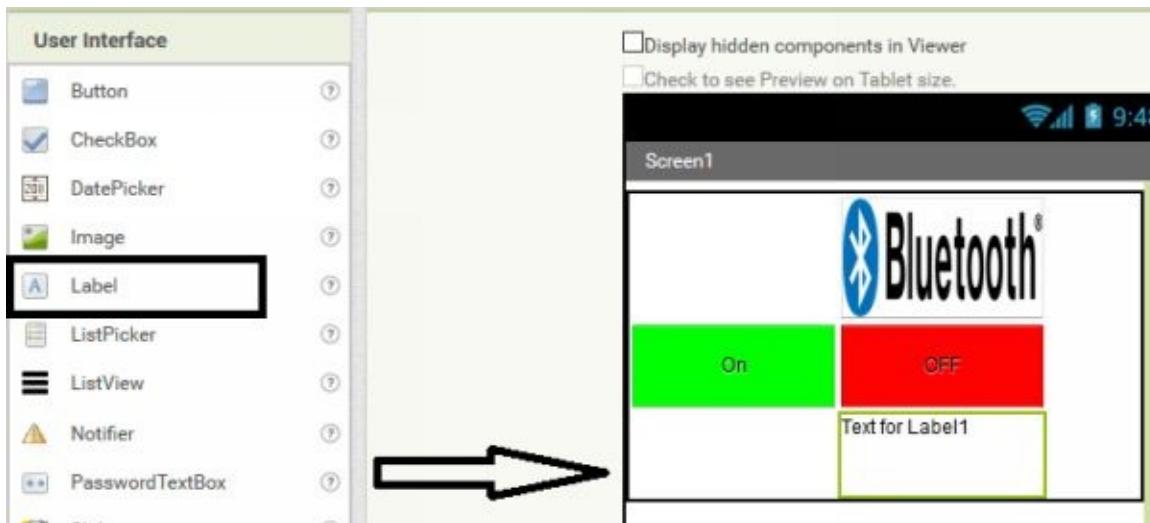
Click on the first button and go the button properties and change the name of the button to ON do the same for the second button by changing the name of the button to OFF.



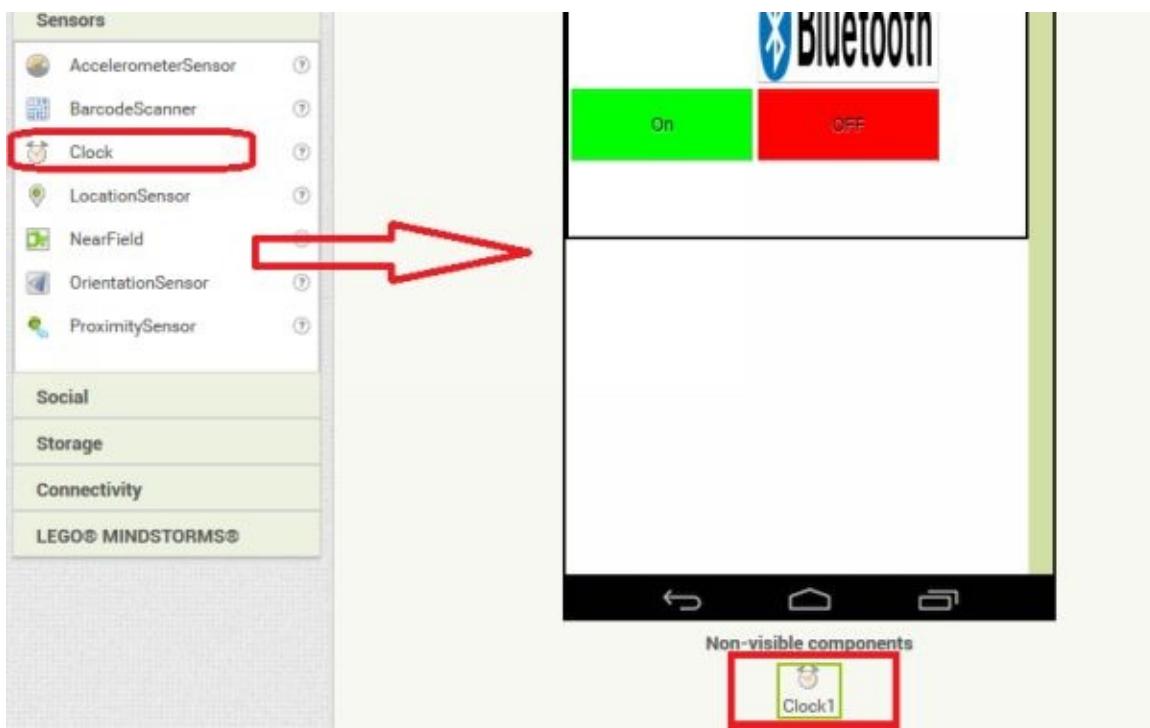
Change button color by changing the background color from properties of the appropriate button.



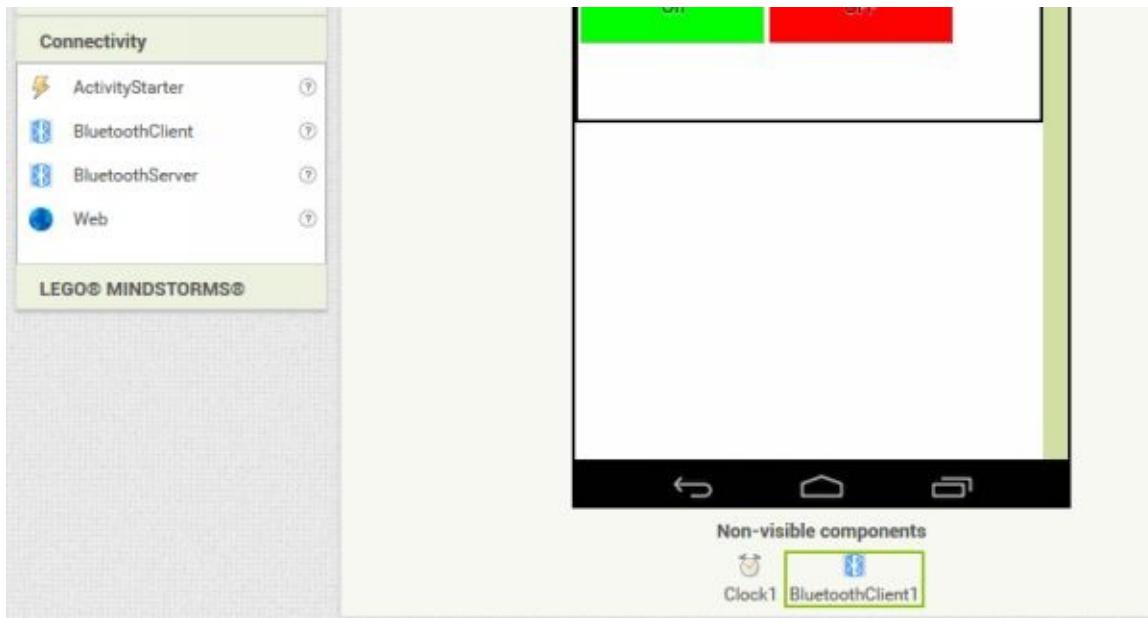
Pick and place Label from user interface and place into the 3rd row of Viewer Screen. Go to properties and remove the Text label and leave this as empty, this label will later on used for indicating whether our app has connected to an Bluetooth or not.



We have almost completed User Interface for our first Android app, we also need some Non-visible components to make this app, the first one is the clock which can pick and placed from the sensor palette



It provides the instant in time using the internal clock on the phone. It can fire a timer at regularly set intervals and perform time calculations, manipulations, and conversions. We are using clock to cause activities to occur at a preset interval.



Last step in this tutorial pick Bluetooth client form the connectivity palette and place it inside the Viewer screen. The Bluetooth client component which allows us to connect to other device and make us to send and receive data's between Bluetooth devices.

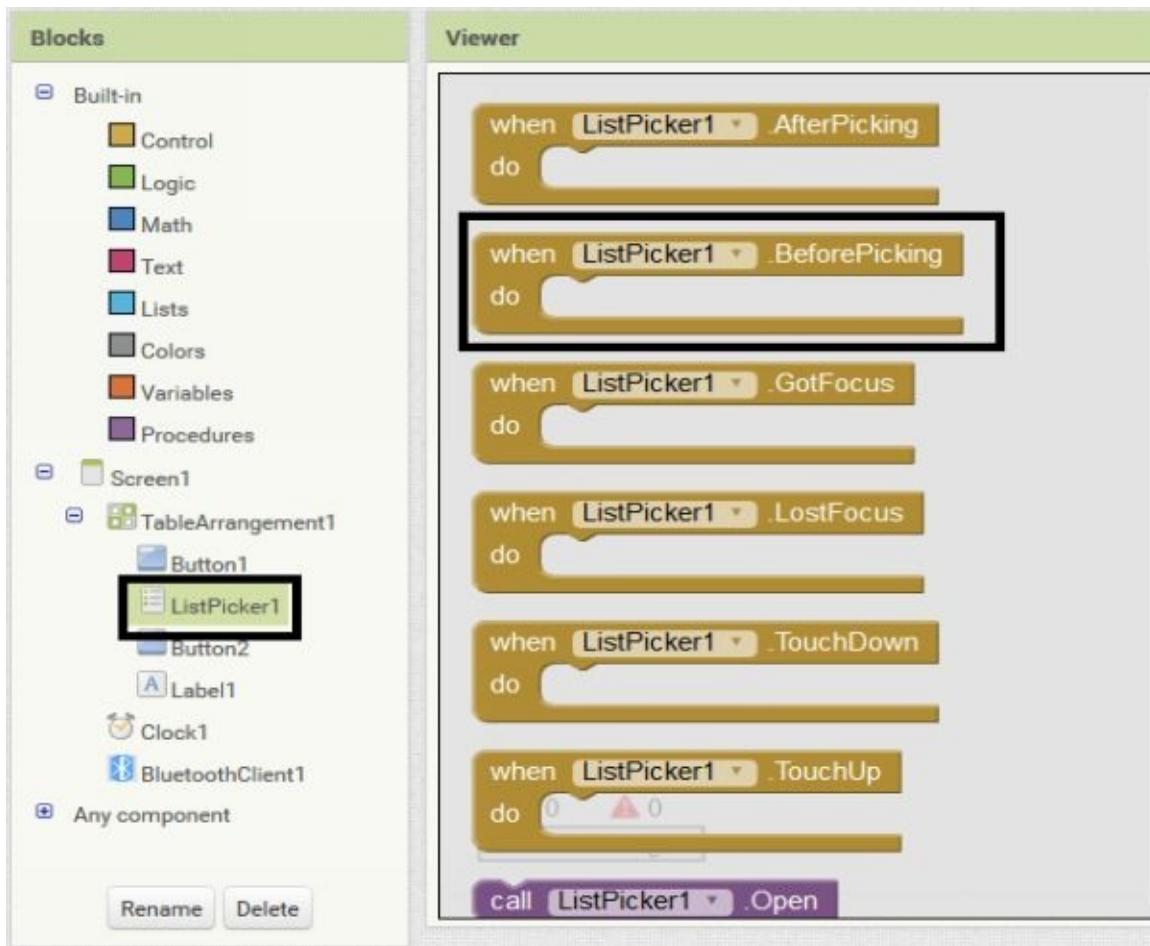
We have completed the user interface design for the first app, now this design requires program to run behind this and do certain functions, In app inventor we connect different blocks to make a program and it's easy for any new user who's having lot of enthusiasm for learning, let's move to programming.



Click on the blocks to get into programming mode.



Click ListPicker1 and select when.listpicker1.BeforePicking block and place into programming window.



After placing the first block and scroll down in the list picker blocks and select set.Listpicker1.element to

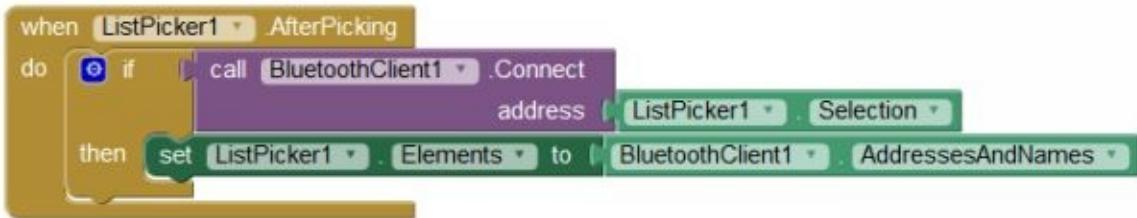
```
when ListPicker1 . BeforePicking
do set ListPicker1 . Elements to
```

Now you may familiarize with blocks and how to pick and place into programming window, complete the blocks programming by going into different blocks associated with different functions and place it on the programming window. Go to Bluetooth client and pick Bluetooth client address and names and complete the first block.

```
when ListPicker1 . BeforePicking
do set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames
```

The first block which explains, when you click the list picker button, it should display the available Bluetooth devices that are already connected to your device previously.

Now complete the second block by going to the different components and pick and place the required block into programming window.



If then block can be pick from the built in functions.



Second block which explains, when you pick a Bluetooth device from the available list the current device that you picked should be used as a current device and Bluetooth address will be stored and the app will communicate with the selected device.

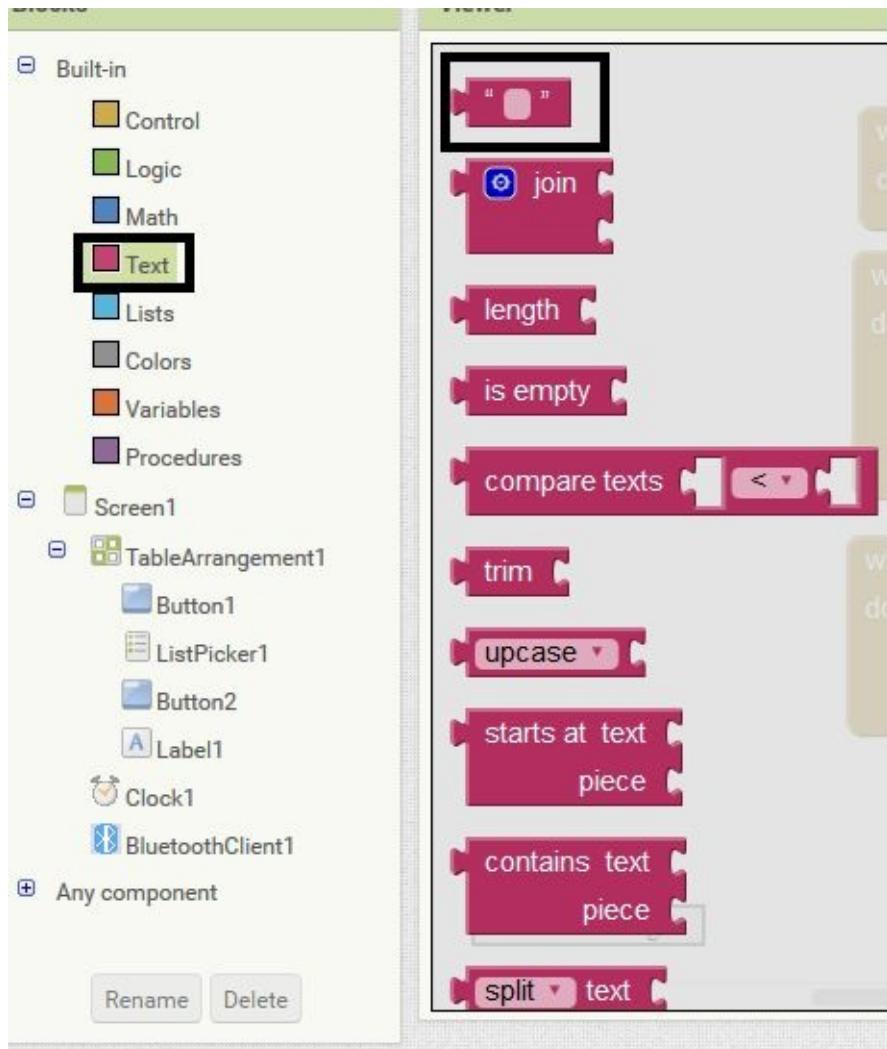


The third block which is placed inside a timer, this block will assign Connected text to Text Label on the third row if the Bluetooth device is connected if the device is not

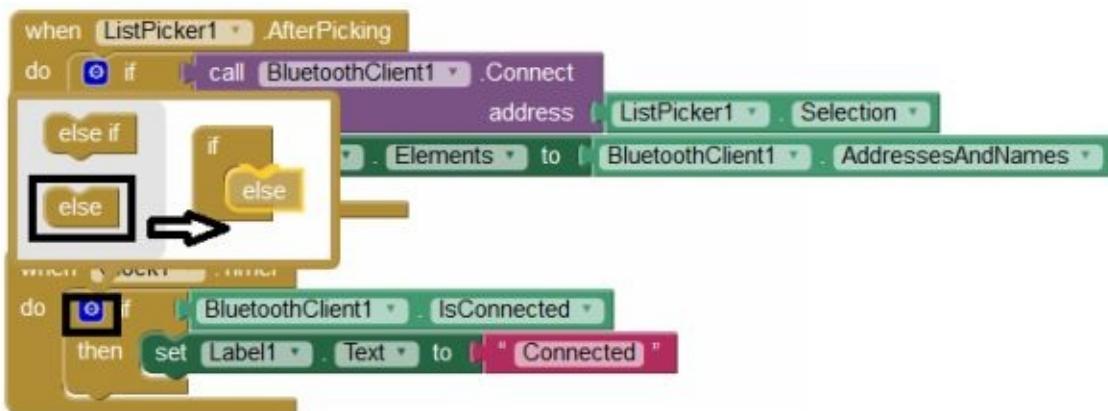
connected Text label will display message as Not Connected.

Complete the block by using the above screen shot.

Text string block can be added from Text in built function

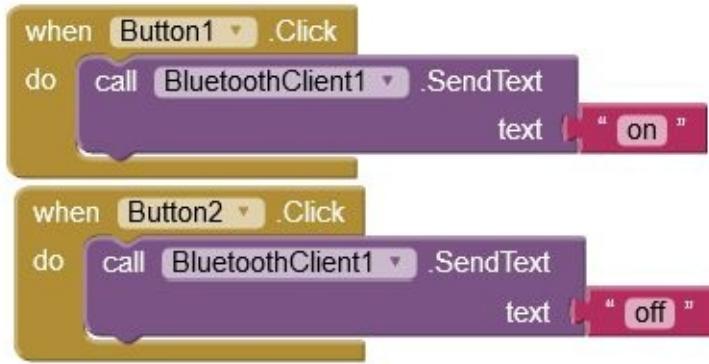


We have already seen how to add If then block from controls, in the above program for screen shot we have added If then else block, which is not available in the controls, we need to place usual if then block and click on the blue icon on the if then block and place the else function to the block.



And we are just few steps away from completing our app. You need to create a control task for the two button we created in the Viewer screen. Complete the code for button

using the screenshot below.



When you press the first button it will send “on” and for the second button it is “off”. We connect to a Bluetooth device attached to Arduino, when Arduino receive these two string we have to program based on these string and make a decision to turn On and Off an LED.

Here's the complete block for the First program to control an LED connected to an Arduino. Check your block with this block and make sure you have not made any mistake when connecting the blocks.

```
when ListPicker1 .BeforePicking
do set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do if call BluetoothClient1 .Connect
    address ListPicker1 . Selection
then set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

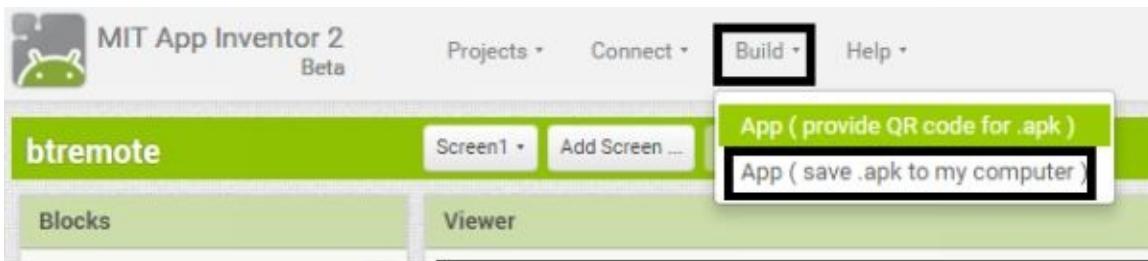
when Clock1 .Timer
do if BluetoothClient1 . IsConnected
then set Label1 . Text to "Connected"
else set Label1 . Text to "Not Connected"

when Button1 .Click
do call BluetoothClient1 .SendText
    text "on"

when Button2 .Click
do call BluetoothClient1 .SendText
    text "off"
```

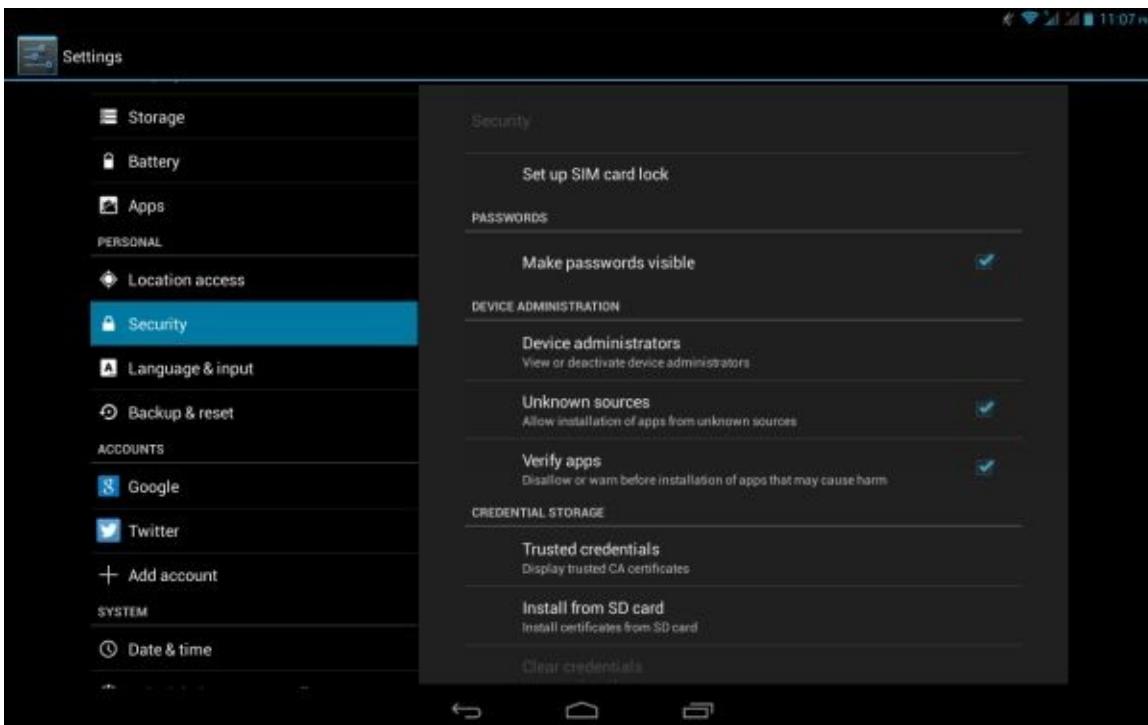
Creating Android App

Convert your block into android app. Click build and select app (save .apk to my computer).



Attach the app to your email and send the Email address which your android device registered to and download the app to your mobile and install it.

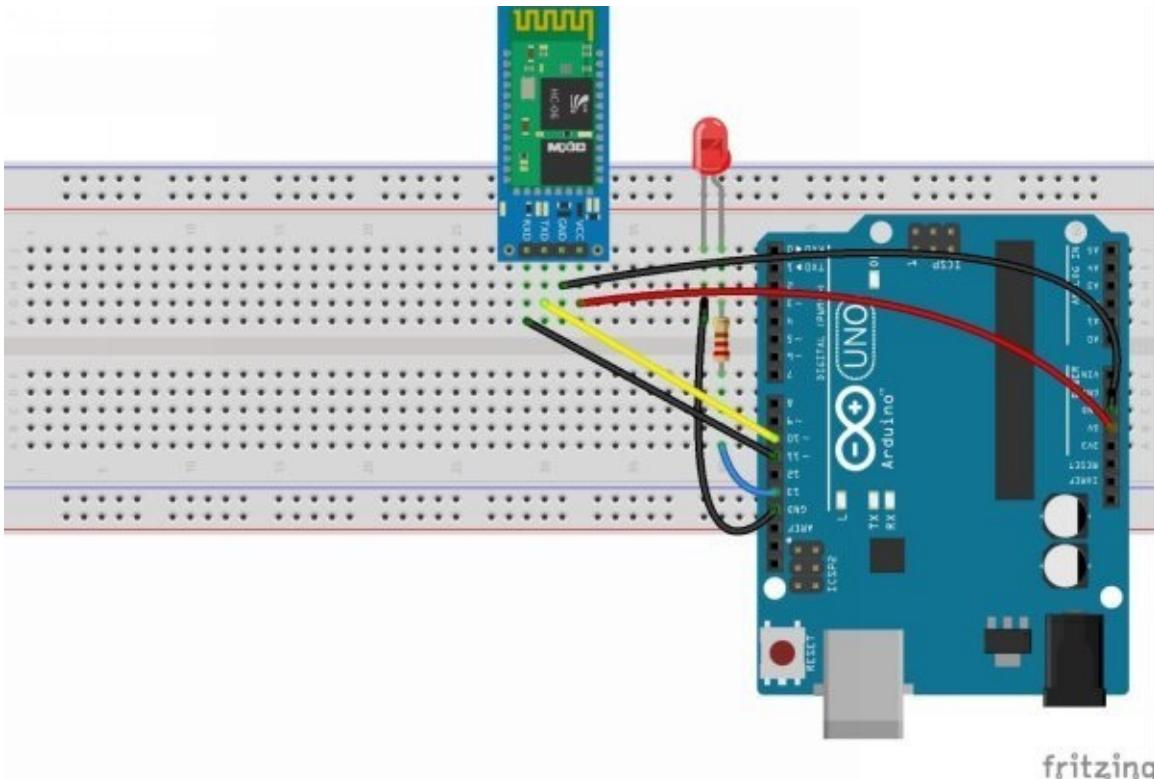
Make sure you have enabled installation of apps from unknown sources, this can be done from your settings.



You need to add HC-05 or HC-06 or whatever model you are using to your android device, connect your Bluetooth device VCC to 5v and Gnd to Ground of Arduino, go to your Bluetooth and turn on your Bluetooth and search for devices, connect to your Bluetooth device using default password, most Bluetooth devices will be '1234' or '0000'.

Connecting Bluetooth and programming Arduino

Now connect your Bluetooth device to Arduino board, connection as follows



Connect your LED to 13th pin of Arduino, if you don't want to connect an LED still you can do this project, by default an LED is connected to the 13th pin of Arduino. Connect TX of Bluetooth to 10th pin and RX of Bluetooth to 11th pin. You can also connect it to the available TX and RX pins that is 0 and 1, when you connect to these two pins you cannot connect your Arduino to your PC or Laptop.

```
// code begin here
#include <SoftwareSerial.h>
SoftwareSerial BT(10, 11); //TX, RX respectively
String state; // string to store incoming message from bluetooth
void setup() {
  BT.begin(9600); // bluetooth serial communication will take place on pin 10 and 11
  Serial.begin(9600); // serial communication to check the data on serial monitor
  pinMode(13, OUTPUT); // LED connected to 13th pin
}
//-----
void loop() {
  while (BT.available()){ //Check if there is an available byte to read
    delay(10); //Delay added to make thing stable
  }
}
```

```

char c = BT.read(); //Conduct a serial read
state += c; //build the string- either “On” or “off”
}
if (state.length() > 0) {
    Serial.println(state);

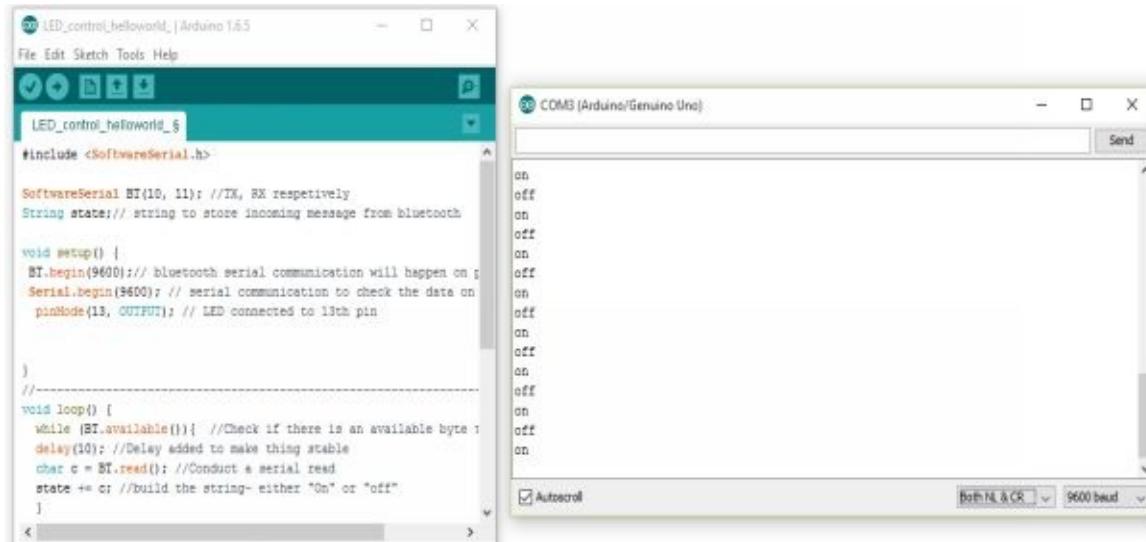
if(state == “on”) // if the received string is on, turn on led connected to the pin 13
{
    digitalWrite(13, HIGH);
}

else if(state == “off”) // if the received string is off, turn off led connected to the pin 13
{
    digitalWrite(13, LOW);
}

state = “”; } } //Reset the variable
// code ends here

```

Upload the above program to Arduino and connect your app and turn on and off the led connected to your Arduino, you can also check the Arduino serial monitor for incoming data.

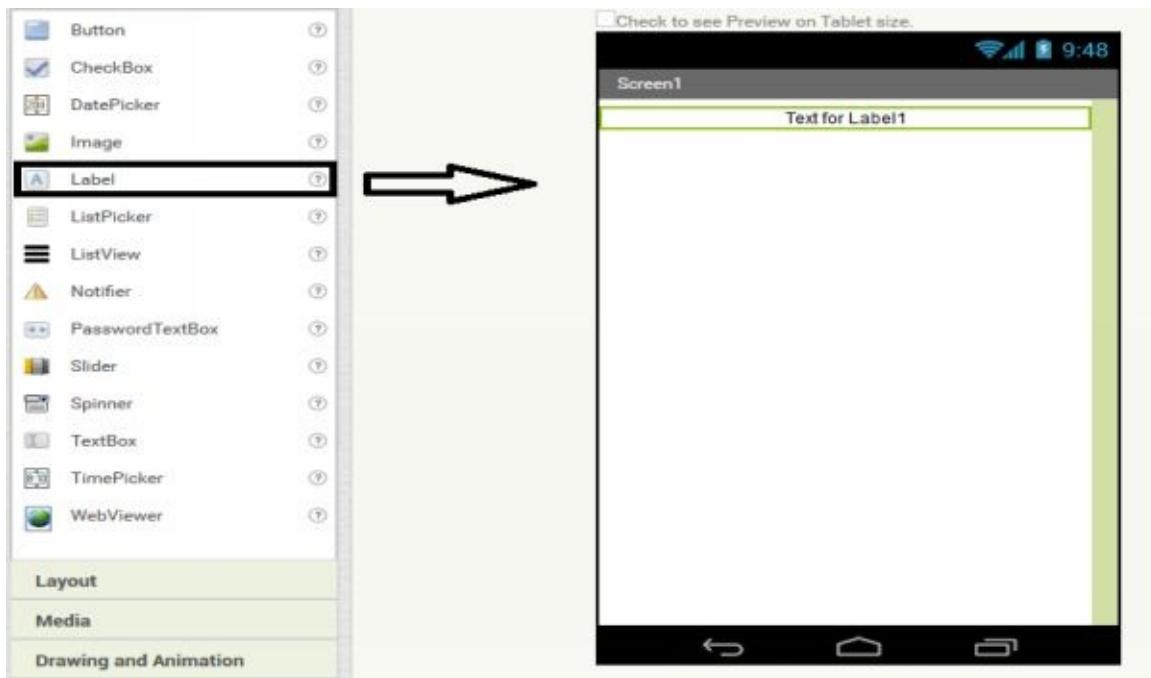


Control LED by Speech recognition

In the previous chapter we have seen how to control an LED connected to Arduino by creating an android app, in this chapter we will see how to control the same LED and the same setup and program that we used in the previous chapter by speech recognition, this app is very simple, all the android phone comes with Google inbuilt speech recognition engine, it need to connect to internet to work properly, when you speak something to your mobile phone it recognize your speech and convert it to text, we simply transfer the text to the Bluetooth device and match the text with predefined text in Arduino, if the text send from our android mobile matches then Arduino executes certain task associated with the text.

Steps to create android app in MIT app inventor

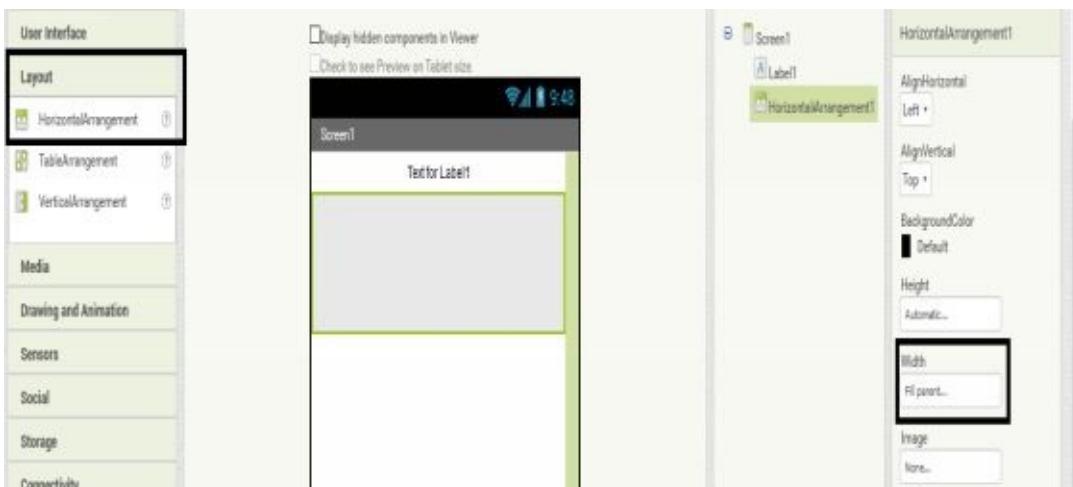
- 1) Go to <http://ai2.appinventor.mit.edu/> and create new project and name the project to your wish.
- 2) Pick label from user interface and place into Viewer screen, This will be used for displaying Bluetooth connectivity, if the Bluetooth is connected , connected text will be displayed if the Bluetooth is not connected , not connected text will display in the label.



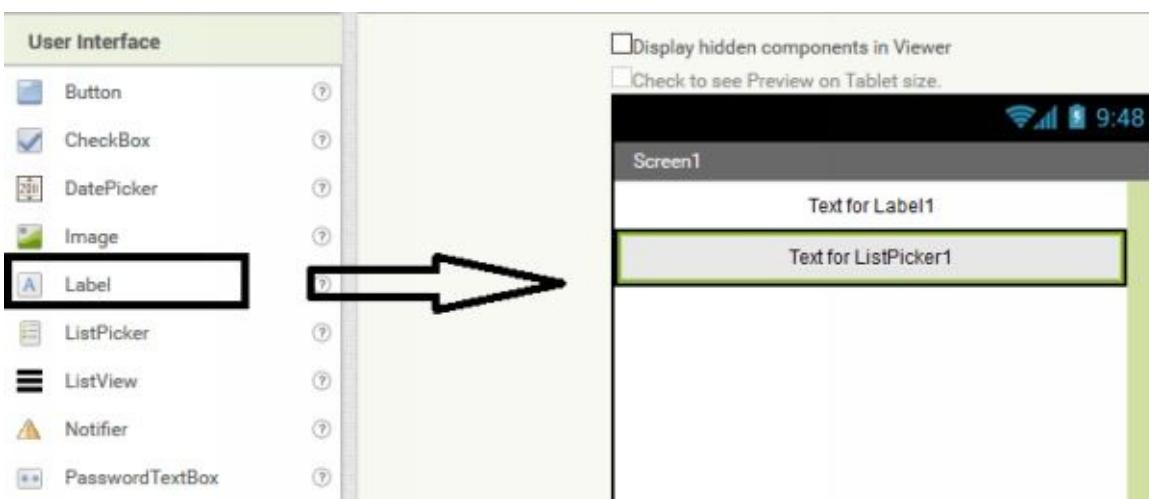
Go to properties of the label and change the width to fill parent and delete the text



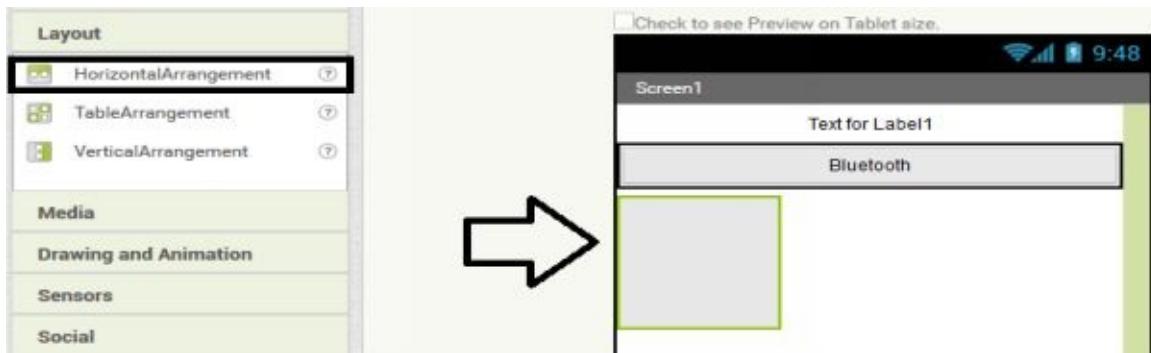
- 3) Pick and place horizontal arrangement from layout and change the properties of width to fill parent.



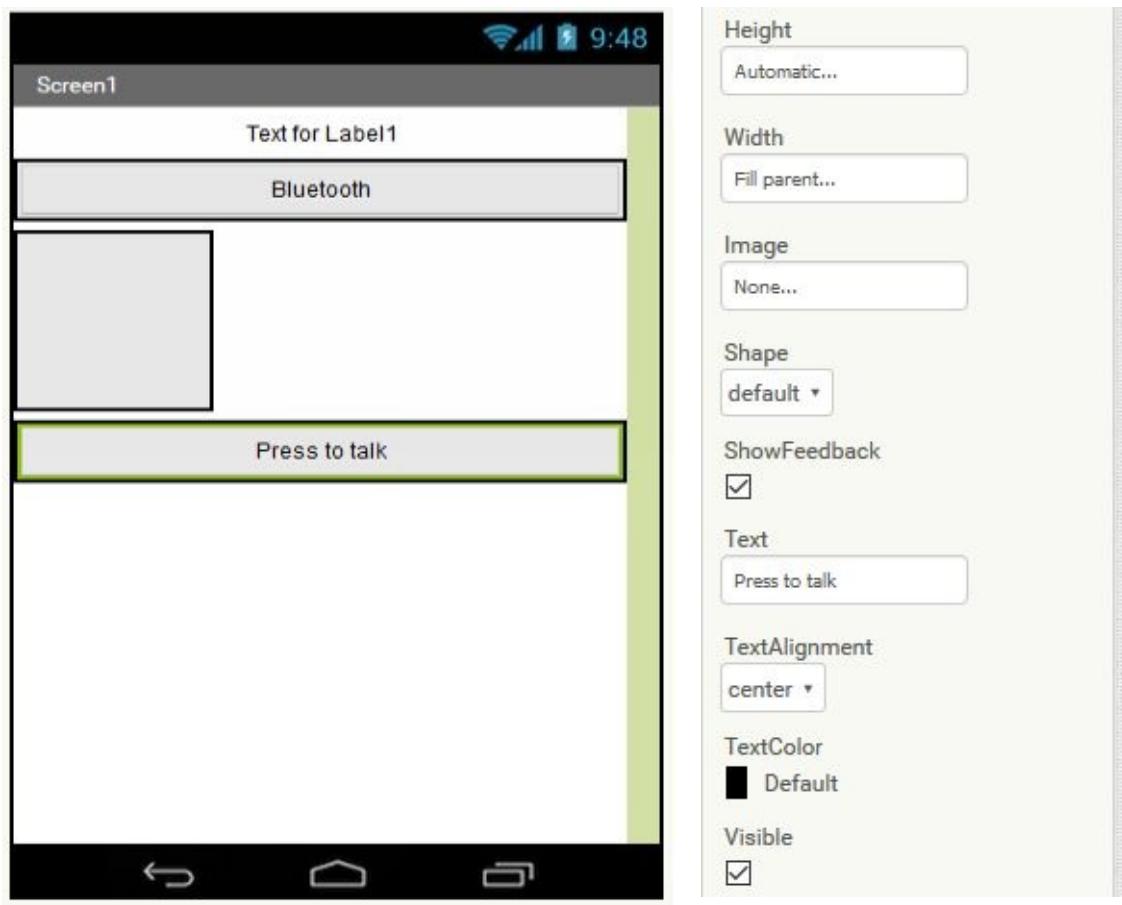
- 4) Pick list picker from user interface and place into the horizontal arrangement box, and change the list picker properties width to “fill parent” and change the text to Bluetooth, you can also add an image to the list picker.



- 5) Pick another horizontal arrangement table and from layout, this table will add a space between the Bluetooth picker and button which will look good instead of the button immediately follows the Bluetooth picker.



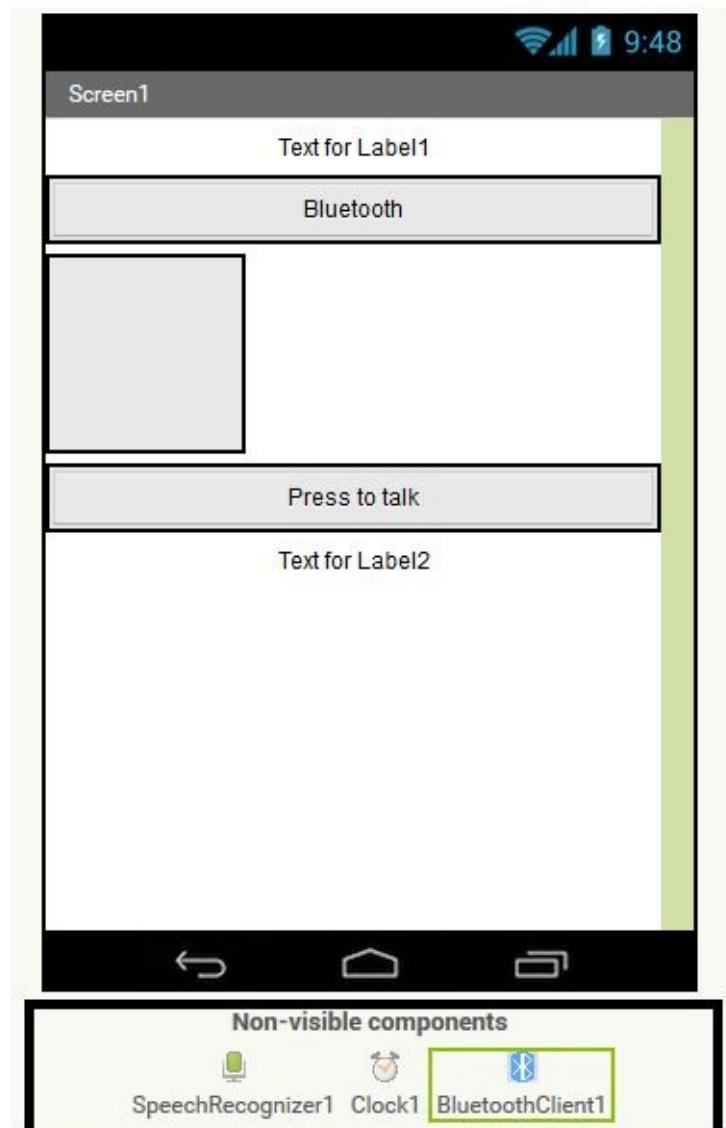
- 6) Pick one more horizontal arrangement and place it below the previous arrangement and change the properties of the present horizontal arrangement width to “fill parent”, pick a button from user interface and place into the horizontal arrangement. Change the button properties width to fill parent and change the text to “ press to talk”, add an image to the button, you can use a google talk icon to make it more presentable and attractive.



- 7) Add a label below press to talk button, this label will be used to display the result from speech recognition edit the properties of label and remove the default text associated with this label and change the width to fill parent and make the text align to centre.



- 8) For the final finishing stage we need to pick and place non visible components to the Viewer, we need to place clock, Bluetooth client, and speech recognizer, pick clock from the sensor palette, Bluetooth client from connectivity palette and speech recognizer from Media palette.



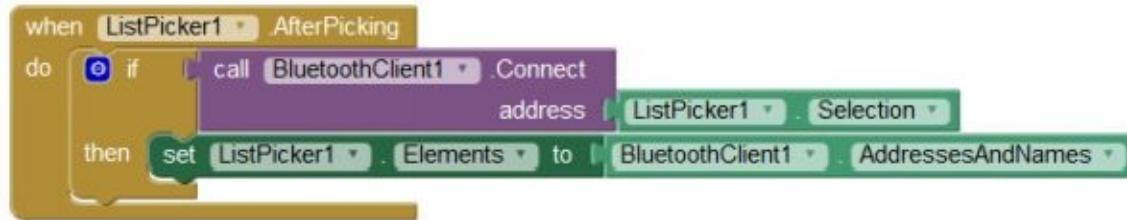
After completing everything in the Designer section it's time to program using blocks, click on the blocks on the top right hand side and move to programming section, we have already discussed about programming section in last chapter, let's directly dive into coding and see whatever the blocks need to be pick and place to make this app to work.

First place the block functions for before picking the listpicker what should happen in the program.



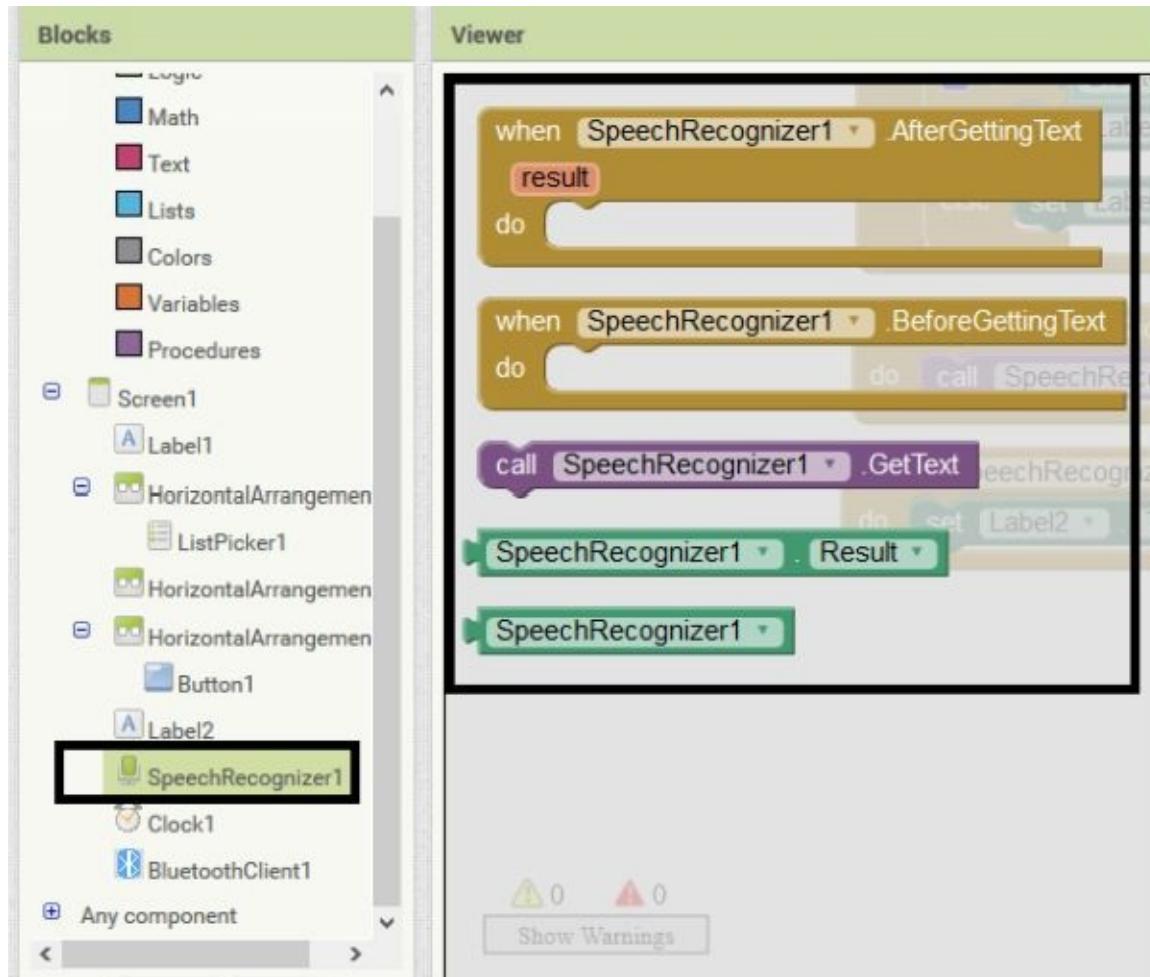
When you place this function, when you press the list picker button it will display available Bluetooth device that has already connected to your Android mobile.

Now, place the block for the second function, what should the app do after connecting to a Bluetooth device



This function will set the address of the Bluetooth device that we selected mostly we select the HC-05 or HC-06 Bluetooth device connected to Arduino.

All the block that is associated with speech recognizer can be find the speech recognizer block. Click on the speech recognizer to get the blocks and choose the blocks by following the screenshot for the program.



Now, we place the third block which will set the text label 1 to connect if the Bluetooth device has successfully connected else it will set to not connected.



Now we will move on the main program, we added a button in the viewer, when someone press that button it should open Google speech recognizer and start listening to you and convert your speech to text.



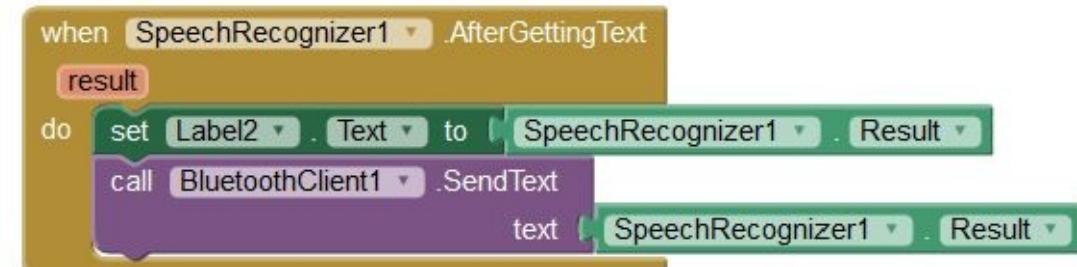
Add the above blocks to start the speech recognizer and get the text.

Now we got the text, it should be stored into label 2 and before storing to label 2, previous text that occupied in the label 2 should clear. Make the block by following below screenshot to do that.

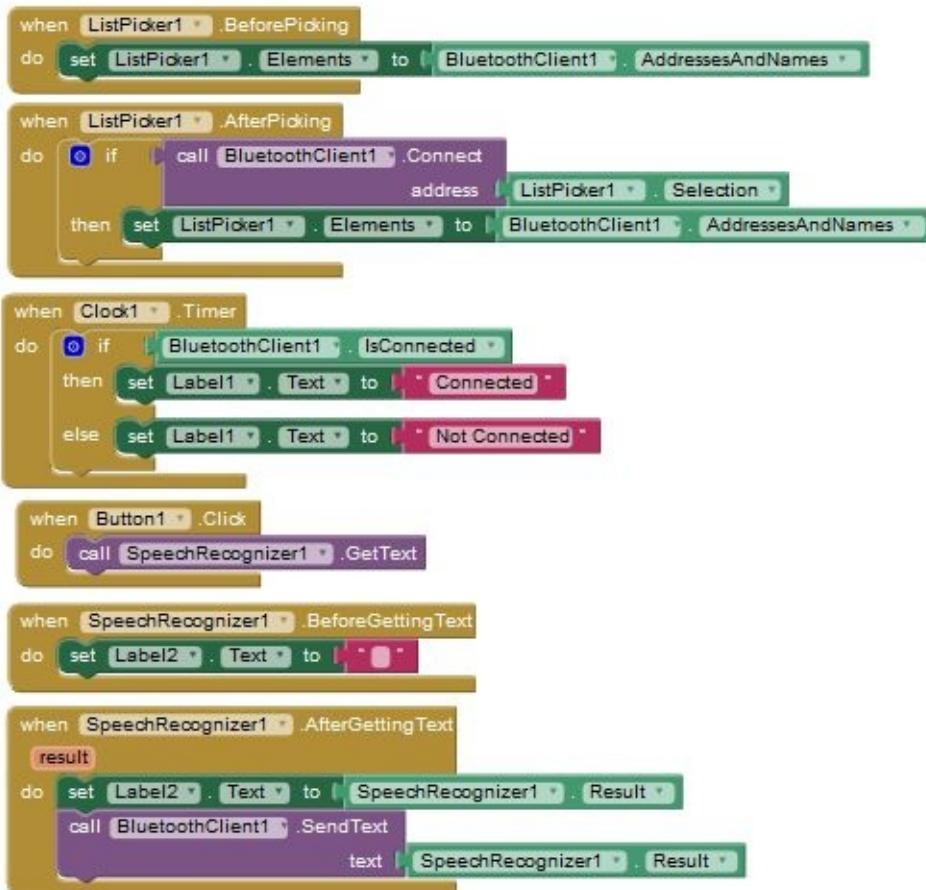


This block will help us to empty the text that previously occupied in label 2.

Now we will add blocks for sending the converted speech to text character as a Bluetooth message and also display the text in label 2.



Above block will send the result as string to the Bluetooth device connected to Arduino, we have already programmed Arduino, so if the text matches to it then Arduino will execute appropriate function that is associated with the text.



Complete program for the speech recognition has given in the above screenshot, that can be used to verify whether you have programmed without any errors or not. Upload the same Arduino program to your Arduino board and start talking to your app, when you say ON the led on the Arduino will goes on and when you say OFF led on the Arduino side will goes off, check this app with your Android device.

Refer [Connecting Bluetooth and programming Arduino](#)

Task

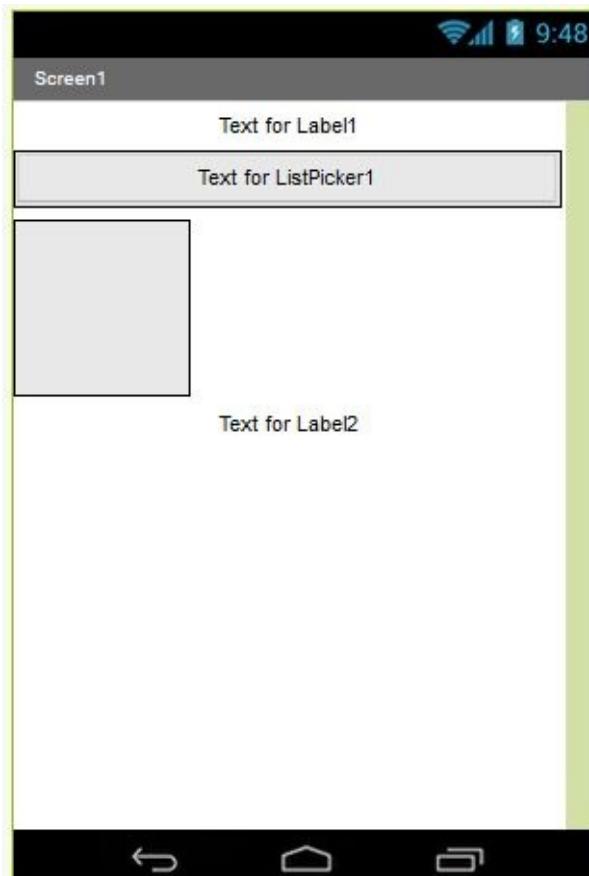
Combine both the app speech recognition and button control into one app and play with your android Arduino.

Control a Servo Motor

In this chapter you will create an app for controlling the servo motor and also program Arduino to receive commands from android mobile and angle the servo motor to the position that is assigned by the android app.

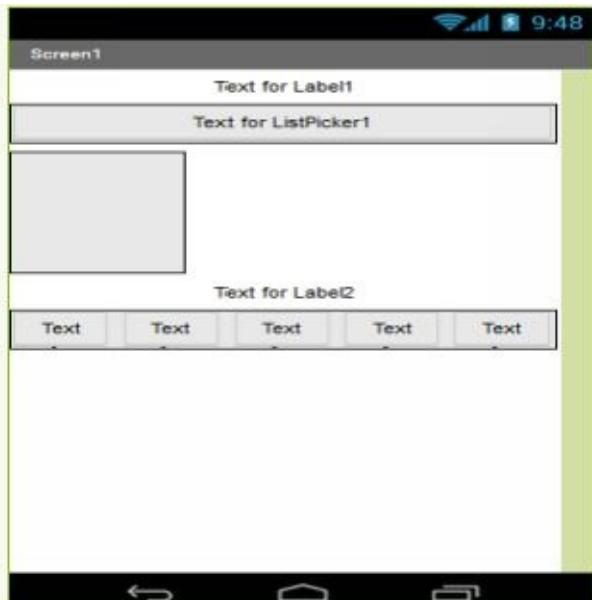
You need a servo motor and Arduino board with Bluetooth module, use any of the servo motor that is available in the market, some motors draws more power and Arduino cannot supply enough power to run the servo motor, in such cases you should connect an external power supply to your Arduino this will make the Arduino board to work smoother during the operation, if you are not supplied with external power Arduino may frequently reset when running in laptop or PC USB power.

- 1) Click on start new project and name your new project as servocontrol or any other you would like for your android app.
- 2) Pick label from user interface and place it in Viewer screen, change the properties of label width to fill parent, and text align to centre.
- 3) Pick horizontal arrangement from layout and place it below Text label-1 and change the properties of horizontal arrangement width to fill parent
- 4) Pick another horizontal arrangement and place it below previous one. This will be used for spacing.
- 5) Pick label from user interface and place it below previous horizontal arrangement we added for spacing.



6) Add another horizontal arrangement below Text label 2 and change its properties width to fill parent.

7) Pick and place a button from user interface and place it inside the horizontal arrangement that created newly, add another 4 buttons to the same horizontal arrangement.



8) Change the button properties from text for button to 0 for the first button, 45 for the second button, 90 for the third button, 135 for the fourth button and 180 for the last button.



9) Pick slider from user interface and place it in Viewer screen below the buttons that have created in the previous step. Change the slider properties Max value to 180 and Min value to 0.

10) Pick and place invisible components, Bluetooth client and clock into Viewer screen.

Designer part is over and the programming part is left over, click on the blocks on the upper right side of the App inventor window.

Complete the first 3 blocks which has repeated in the first 2 chapters, if you have skipped these chapters and directly launched into this chapter, please go through first or second chapter to make sure how this code works and what you should do to complete the below code given in the screen shot.

```

when ListPicker1 .BeforePicking
do set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

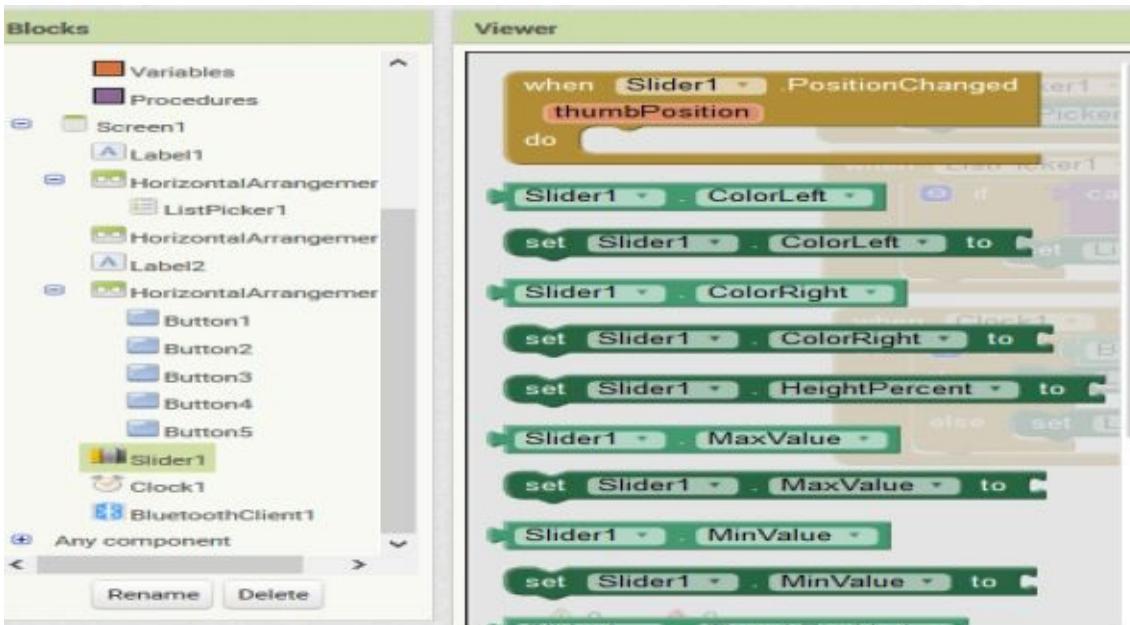
when ListPicker1 .AfterPicking
do if call BluetoothClient1 .Connect
address ListPicker1 . Selection
then set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when Clock1 .Timer
do if BluetoothClient1 . IsConnected
then set Label1 . Text to "Connected"
else set Label1 . Text to "Not Connected"

```

Above code will connect to the Bluetooth device that is already paired with your android phone and if the device is connected then label 1 text changes to connected else label 1 will display as not connected.

Function related to slider block can be find in the slider check the image below to know how to access blocks related to slider.



Drag and drop the below screenshot for the blocks in the blocks window of Mit app inventor, this block will send the number when you change the slider position in your android app, and the current slider position will be display on your label 2.

```

when Slider1 .PositionChanged
thumbPosition
do set Label2 . Text to round Slider1 . ThumbPosition
call BluetoothClient1 .Send1ByteNumber
number round Slider1 . ThumbPosition

```

Round function will be new to you and this function is introduce in this chapter, the round

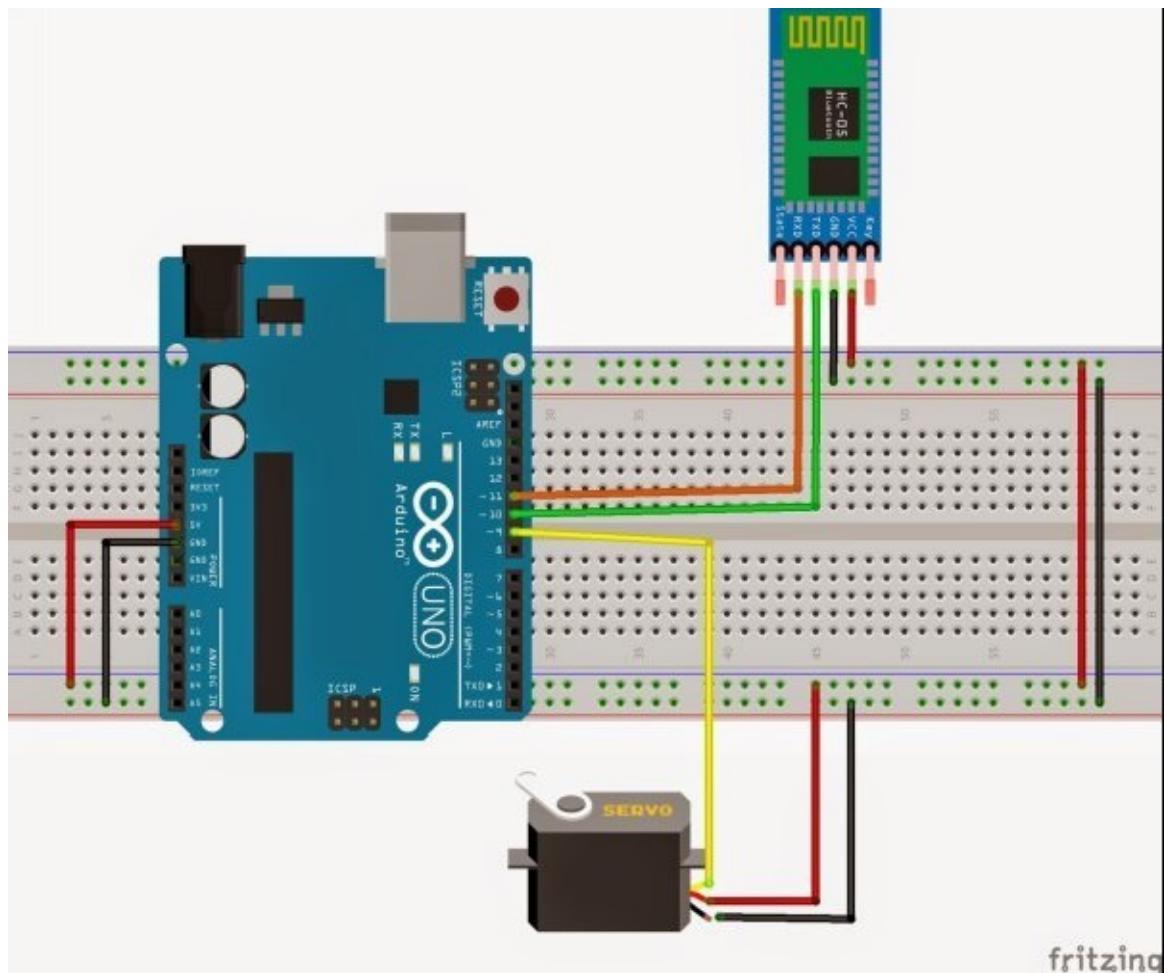
function help us to round a number all the number will be presented as a whole number. This block is available in the built in math block.



Complete the below block for sending numbers when each button is pressed, we have created five buttons in the Viewer screen and each button has assigned a number, when we press a button the number assigned to the number should send to the connected Bluetooth device. Block to give number input can be find in the math built in function.

Arduino code for servo control

In this section we will know how to connect the servo motor and Bluetooth to your Arduino.



Connect your Bluetooth TX pin to the 10th pin of Arduino and RX pin to the 11th pin of Arduino, most of the servo motor comes with 3 wires and check for your servo motor wire specification with the manufacturer sometime servo motor comes with different wire colors, connect the servo signal wire to the 9th pin of Arduino.

If your Arduino board reset frequently when you connecting your app, this might be due to low power from the Arduino side, to eliminate this error you can supply external power supply to your Arduino which can make things stable.

Write the below code and all the explanation for this code is given as a comment next to each line of Arduino program, try to type the code and know what each code represents and this code can be correlate with the android program. Don't copy and paste without understanding what is actually happening inside the code.

Arduino program for servo motor:

```
// code begins

#include <SoftwareSerial.h> // TX RX software library for bluetooth
#include <Servo.h> // servo library

Servo myservo; // servo name

int bluetoothTx = 10; // bluetooth tx to 10 pin
int bluetoothRx = 11; // bluetooth rx to 11 pin

SoftwareSerial bluetooth(blueoothTx, blueoothRx);

void setup()
{
    myservo.attach(9); // attach servo signal wire to pin 9
    //Setup usb serial connection to computer
    Serial.begin(9600);

    //Setup Bluetooth serial connection to android
    bluetooth.begin(9600);
}

void loop()
{
    //Read from bluetooth and write to usb serial
    if(blueooth.available()> 0 ) // receive number from bluetooth
    {
        int servopos = bluetooth.read(); // save the received number to servopos
        Serial.println(servopos); // serial print servopos current number received from bluetooth
        myservo.write(servopos); // roate the servo the angle received from the android app
    }
}

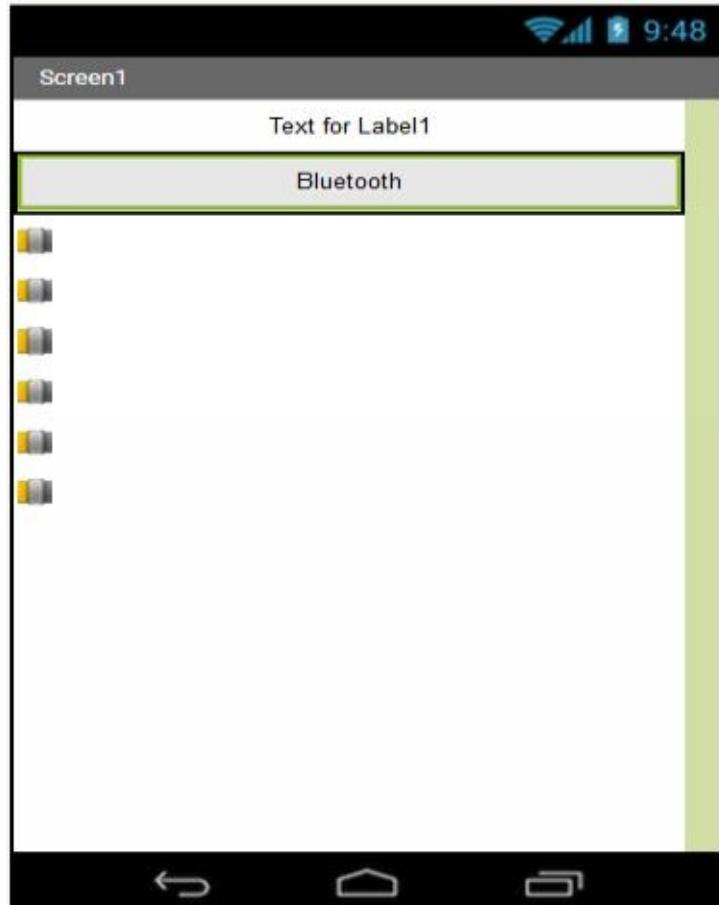
// code ends here.
```

Control Multiple Servo Motors

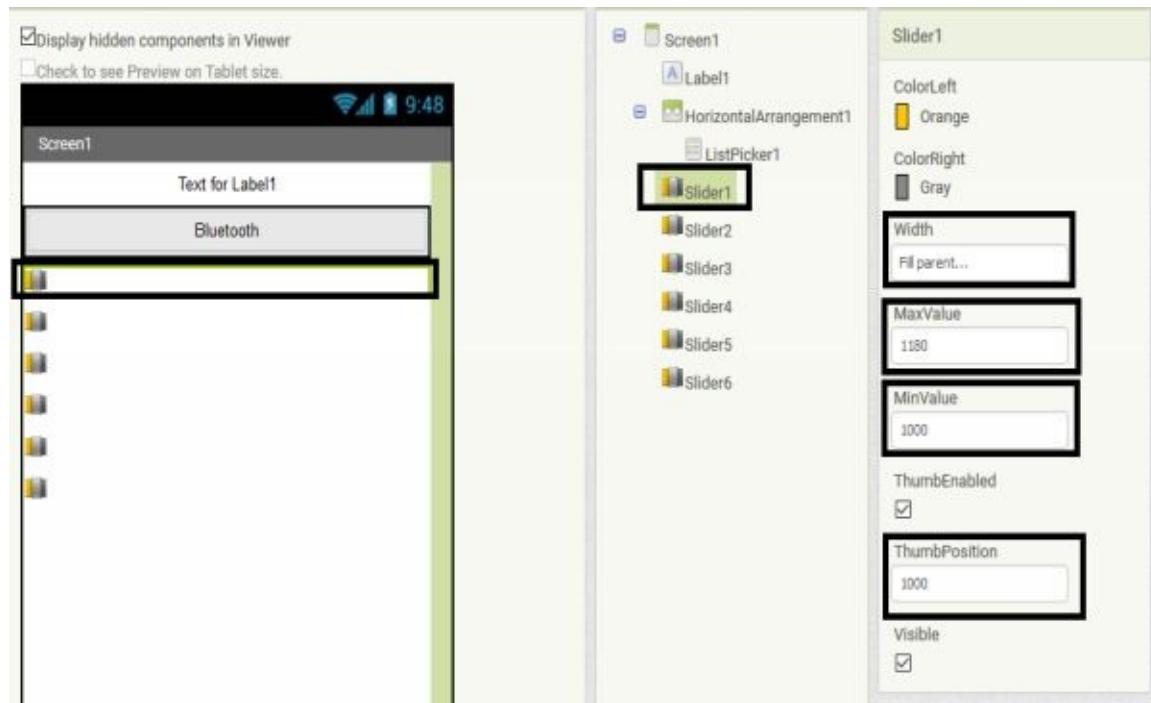
In this chapter we will know how to control many servo motors, this chapter will resemble everything what we did in the previous chapter for controlling a single servo motor, in this chapter we will concentrating on creating six slider to control six servo motor connected on the Arduino side. All the codes are same for android app, on the Arduino side there will be plenty of change in the programming. It's very easy to control a single servo motor the reason is Arduino can easily accept 8bit number, when we created a single servo slider we assigned the data value from 0 to 180 and it falls below 255(8 bit) so it becomes easy for us , when we want to control more servo we need to assign number more than 255 so it more than 8 bit, and still we don't have problem on android app side it just sending numbers from android app on the Arduino side we have to receive these number and do some calculation and make the number to do certain action.

Let's begin with creating Viewer screen for this app.

- 1) Start a new project on your MIT app inventor and name your project as Mutliple_servo_control
- 2) Start with placing a text label for displaying Bluetooth connectivity status
- 3) Place horizontal arrangement and change its properties width to fill parent
- 4) Place listpicker from user interface and place it inside horizontal arrangement
- 5) Place a slider from user interface below the horizontal arrangement
- 6) Place five slider one after another.



Select slider 1 and change its properties width to fill parent and change maximum value to 1180 and minimum value to 1000 and also thumb position to 1000



Select second slider and change its properties width to fill parent and change maximum value to 2180 and minimum value to 2000 and also thumb position to 2000.

Select third slider and change its properties width to fill parent and change maximum value to 3180 and minimum value to 3000 and also thumb position to 3000.

Select fourth slider and change its properties width to fill parent and change maximum

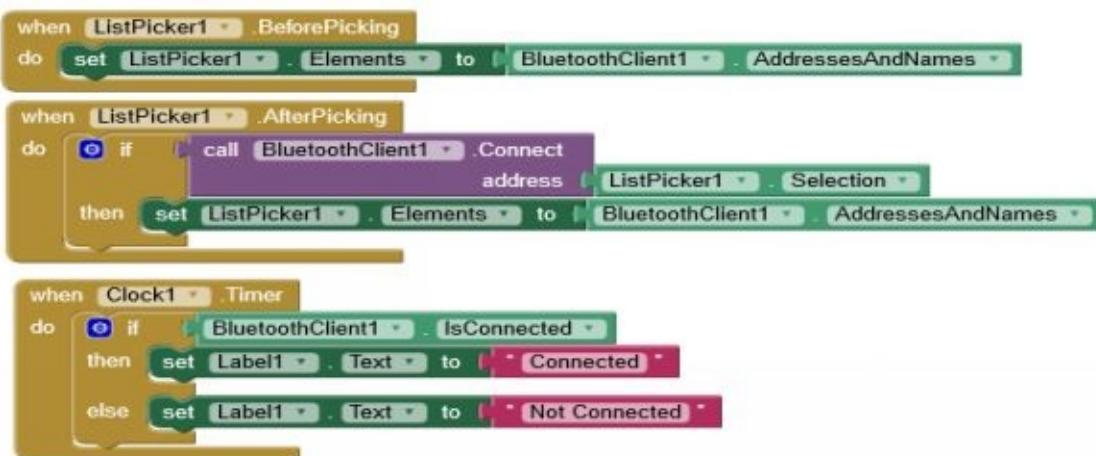
value to 4180 and minimum value to 4000 and also thumb position to 4000.

Select fifth slider and change its properties width to fill parent and change maximum value to 5180 and minimum value to 5000 and also thumb position to 5000.

Select last slider and change its properties width to fill parent and change maximum value to 6180 and minimum value to 6000 and also thumb position to 6000

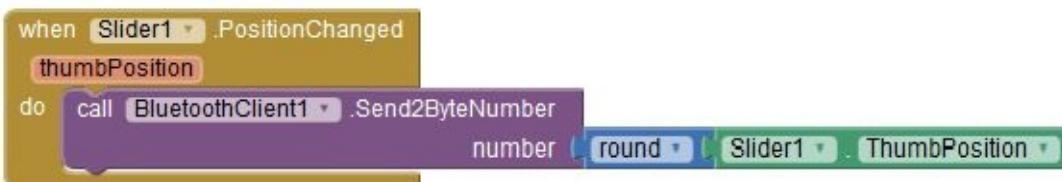
Pick and place invisible components, Bluetooth client and clock into Viewer screen.

Complete the first 3 blocks which has repeated in the first 2 chapters, if you have skipped these chapters and directly launched into this chapter, please go through first or second chapter to make sure how this code works and what you should do to complete the below code given in the screen shot.



Above code will connect to the Bluetooth device that is already paired with your android phone and if the device is connected then label 1 text changes to connected else label 1 will display as not connected.

Complete the block for first slider, whenever slider position changes, these position will send as a 2 byte number to the connected Bluetooth device.



Round block can be pick from Math built in function.

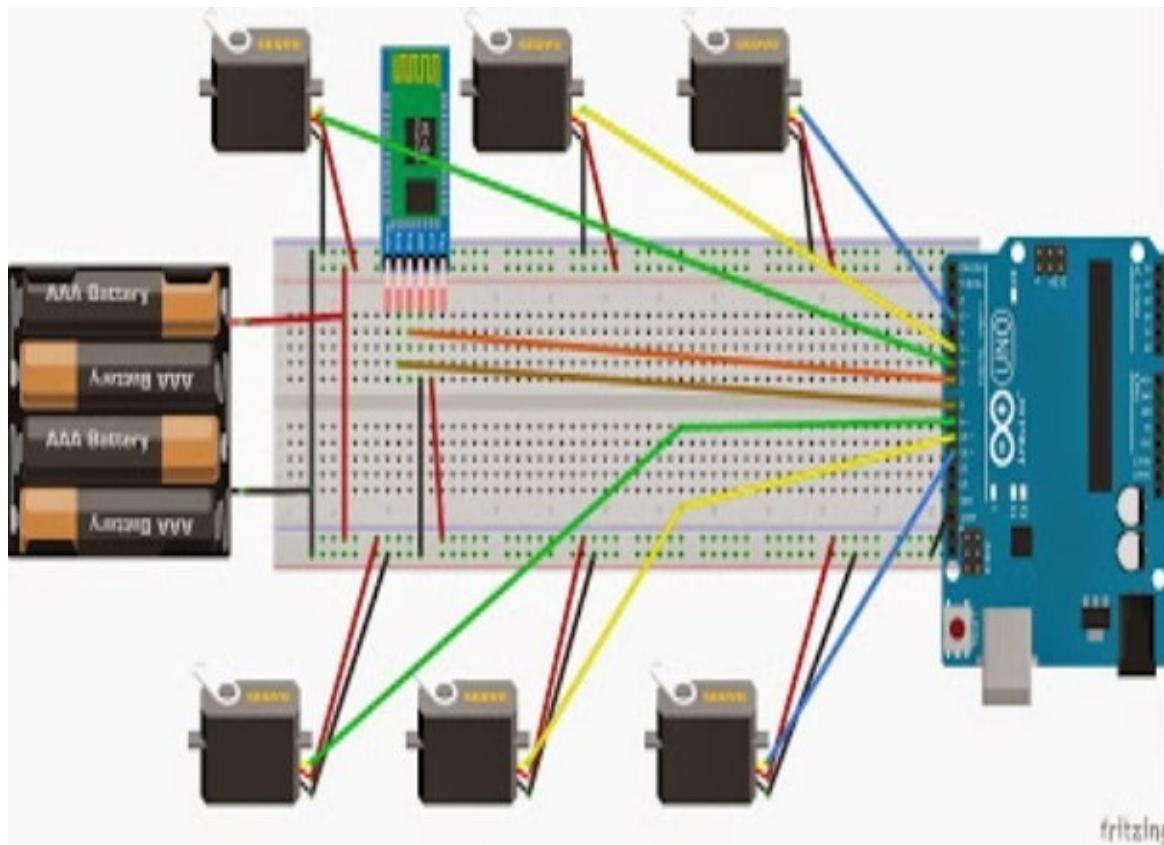
Continue the same block which shown in the above screenshot for other sliders



Programming for the app has completed now build the code and transfer the app to your android mobile and install the app, make sure you have enabled your setting for installing app from unknown source.

Arduino Code Multiple servo

When come's to Arduino code this is quite complicated than our previous programs, this code will receive number from android app, the number from android app is a 2 byte number, so the Arduino will receive these number as 2 bytes and later on add the first and second byte and make it as a single digit number.



When you using servo motor with Arduino board you should use external power supply to ensure proper working of this board, Arduino board cannot supply enough power to these servo motors if it has connected to PC or Laptop.

Connect Tx of Bluetooth to 7th pin of Arduino and Rx of Bluetooth to 8th pin of Arduino.

Connect first servo signal wire to 9th pin of Arduino

Second servo signal wire to 10th pin of Arduino

Third servo signal wire to 11th pin of Arduino

Fourth servo signal wire to 3rd pin of Arduino

Fifth servo signal wire to 5th pin of Arduino

Sixth servo signal wire to 6th pin of Arduino

// code starts here

```
#include <SoftwareSerial.h>
```

```
#include <Servo.h>
```

```
Servo myservo1, myservo2, myservo3, myservo4, myservo5, myservo6;
```

```
int bluetoothTx = 7;
```

```
int bluetoothRx = 8;  
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);  
  
void setup()  
{  
    myservo1.attach(9); // attach servo1 to 9th pin  
    myservo2.attach(10); // attach servo 2 to 10th pin  
    myservo3.attach(11); // attach servo 3 to 11th pin  
    myservo4.attach(3); // attach servo 4 to 3rd pin  
    myservo5.attach(5); // attach servo 5 to 5th pin  
    myservo6.attach(6); // attach servo 6 to 6th pin  
    //Setup usb serial connection to computer  
    Serial.begin(9600);  
  
    //Setup Bluetooth serial connection to android  
    bluetooth.begin(9600);  
}  
  
void loop()  
{  
    //Read from bluetooth and write to usb serial  
    if(bluetooth.available()>= 2 ) // receive two byte numbers  
    {  
        unsigned int servopos = bluetooth.read(); // receive first byte  
        unsigned int servopos1 = bluetooth.read(); // receive second byte  
        unsigned int realservo = (servopos1 *256) + servopos; // add first and second byte to  
        make single digit number  
        Serial.println(realservo);  
        // if number falls between 1000 to 1180 it is first servo  
        if (realservo >= 1000 && realservo <1180){ // check for the number  
            int servo1 = realservo;  
            servo1 = map(servo1, 1000,1180,0,180);  
            myservo1.write(servo1);  
        }  
    }  
}
```

```
Serial.println("servo 1 ON");
delay(10);

}

// if number falls between 2000 to 2180 it is second servo
if (realservo >=2000 && realservo <2180){

int servo2 = realservo;
servo2 = map(servo2,2000,2180,0,180);
myservo2.write(servo2);
Serial.println("servo 2 On");
delay(10);

}

// if number falls between 3000 to 3180 it is third servo
if (realservo >=3000 && realservo < 3180){

int servo3 = realservo;
servo3 = map(servo3, 3000, 3180,0,180);
myservo3.write(servo3);
Serial.println("servo 3 On");
delay(10);

}

// if number falls between 4000 to 4180 it is fourth servo
if (realservo >=4000 && realservo < 4180){

int servo4 = realservo;
servo4 = map(servo4, 4000, 4180,0,180);
myservo4.write(servo4);
Serial.println("servo 4 On");
delay(10);

}

// if number falls between 5000 to 5180 it is fifth servo
if (realservo >=5000 && realservo < 5180){
```

```
int servo5 = realservo;  
servo5 = map(servo5, 5000, 5180,0,180);  
myservo5.write(servo5);  
Serial.println("servo 5 On");  
delay(10);  
}  
  
// if number falls between 6000 to 6180 it is sixth servo  
if (realservo >=6000 && realservo < 6180){  
int servo6 = realservo;  
servo6 = map(servo6, 6000, 6180,0,180);  
myservo6.write(servo6);  
Serial.println("servo 6 On");  
delay(10);  
}  
}  
}
```

// code ends here

Upload the above program to your Arduino board and connect to your android app, you can check the serial monitor for the data receiving from the android app.

RGB Color Slider

In this chapter we will use three slider and mix color using RED slider, GREEN slider and BLUE slider. When we mix RGB we get a new color, this color will be displayed on our app as well as the color data will send to a Bluetooth device connected to Arduino, the same color will be applied to an RGB LED connected to Arduino on the other side.

Activity starter

This activity starter will be useful when the app initialize if we want to execute certain function in android phone we can make use of this activity starter, we will look in this chapter how to use this activity starter to remind us to turn on our Bluetooth if it is disabled during the opening of the app in the android device.

- 1) Pick a Label and place it on the viewer screen, change its properties width to fill parent and change the text alignment to centre.
- 2) Pick a horizontal arrangement from layout and place it below Text label and change its properties width to fill parent.
- 3) Place a list picker from user interface and change its name from properties to Bluetooth and also place a button next to it and name it to disconnect in this chapter we use a button to disconnect from Bluetooth device when we press the button.

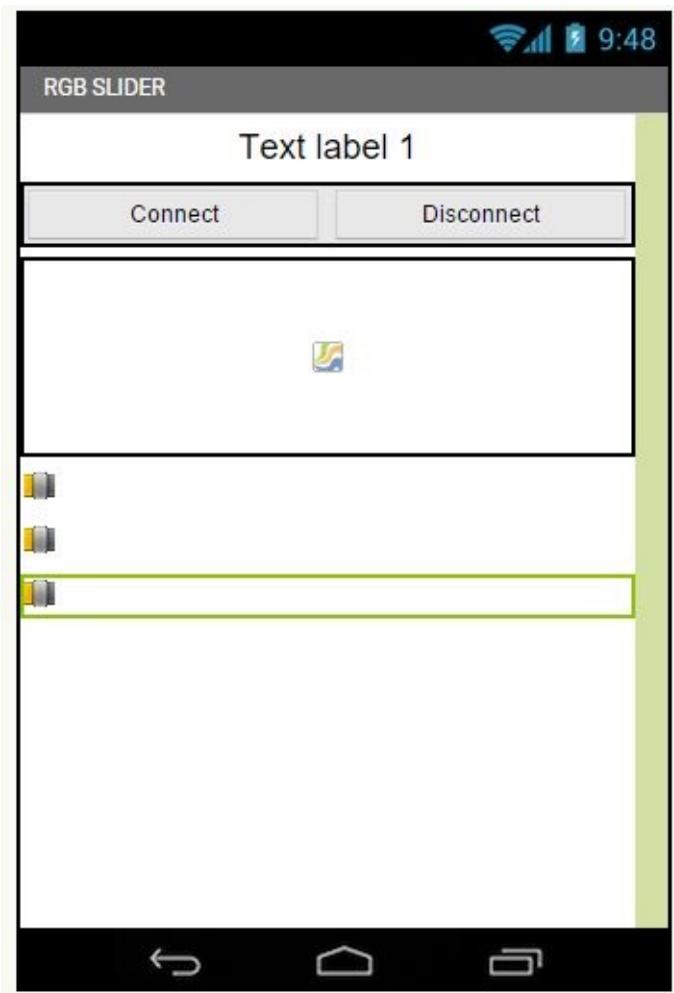


- 4) Pick canvas from drawing and animation and place it below horizontal arrangement and change the properties of width to fill parent and height to 100 pixels.

5)

Pick 3 slider and place

it one after another.

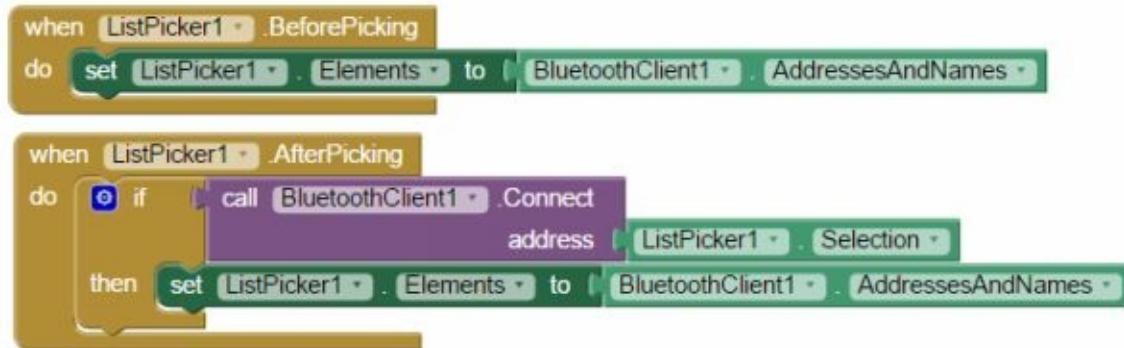


Change the properties of slider as follows

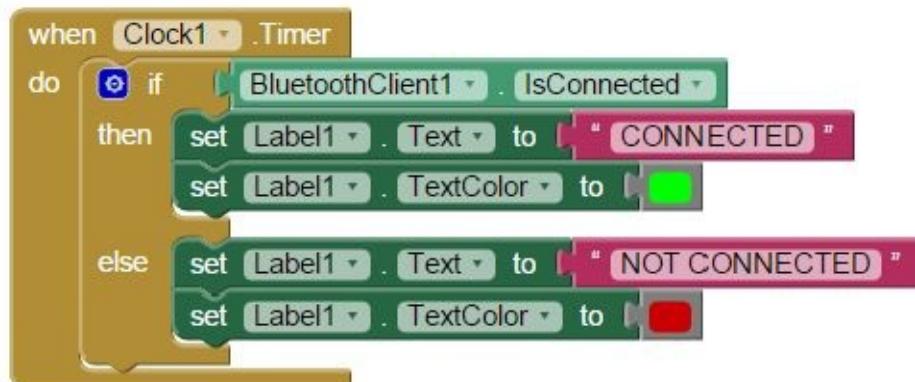
Place non visible components Bluetooth Client, Clock and Activity starter, activity starter can be placed from the connectivity palette.



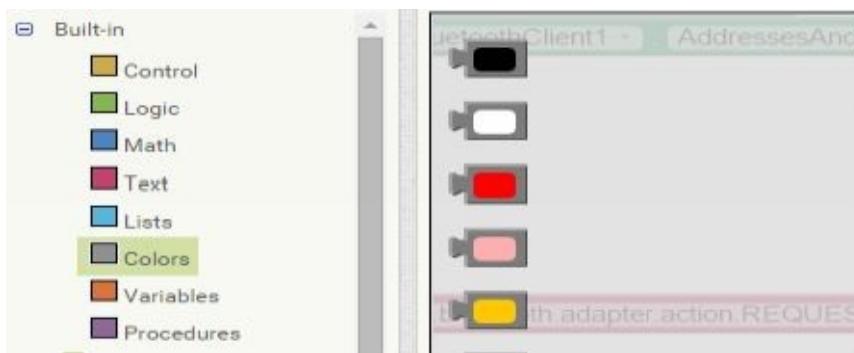
Designer part for this app has completed move on to the blocks for programming this app, click on the blocks to move to the programming window.



Program the RGB slider chooser blocks for connecting to a Bluetooth device, above code will show the list of available Bluetooth devices that you paired already and once you connect to a device this will save the address of the Bluetooth device and connect to it.



Complete the above blocks to display whether the Bluetooth devices has successfully connected or not. Color to the block can be added from the built in colors function.

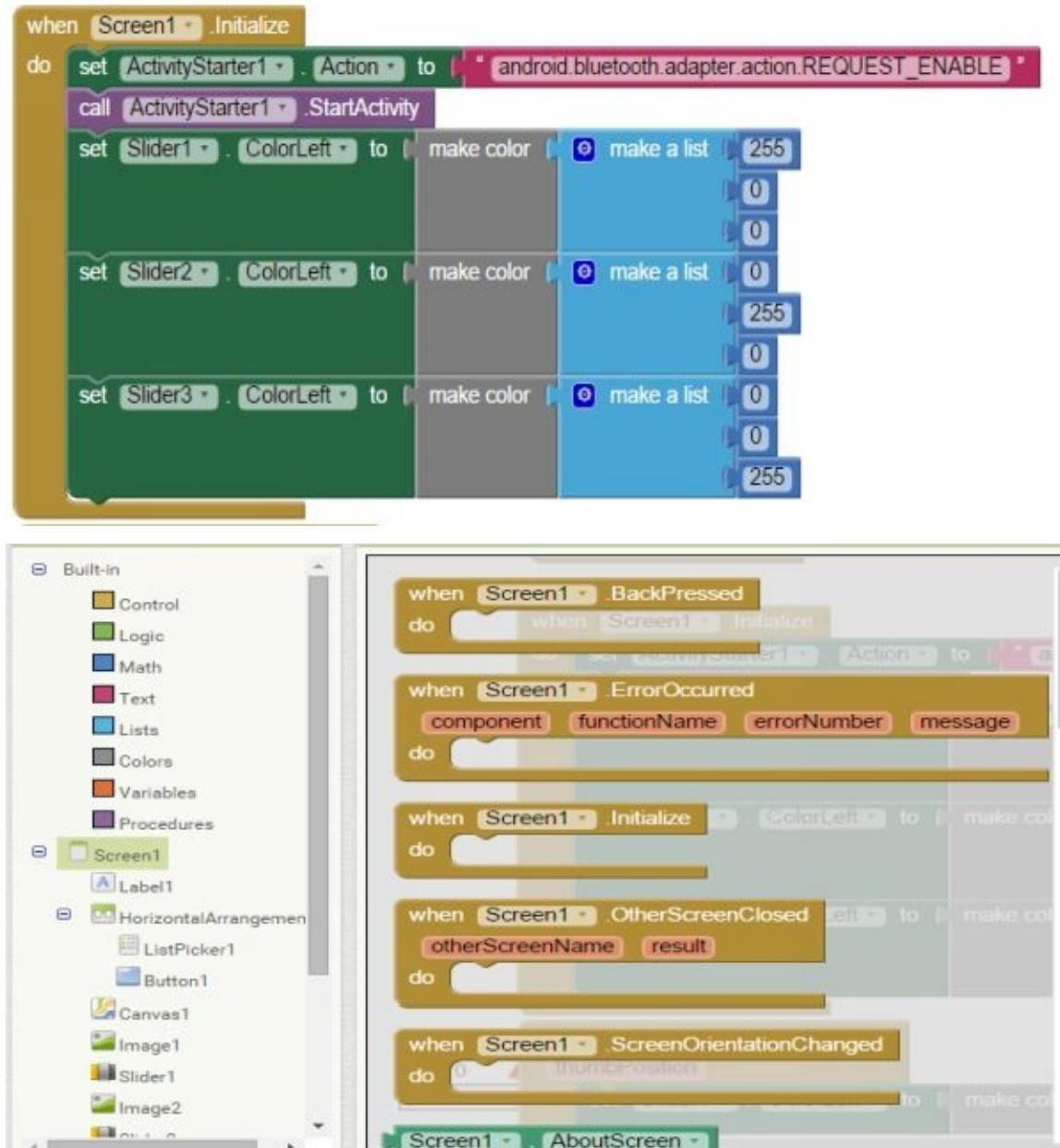


Complete the below code to disconnect from the Bluetooth device that has currently connect to, when you click the app it will disconnect the communication.



When the screen initialize, as we discussed already if the Bluetooth is not currently turned

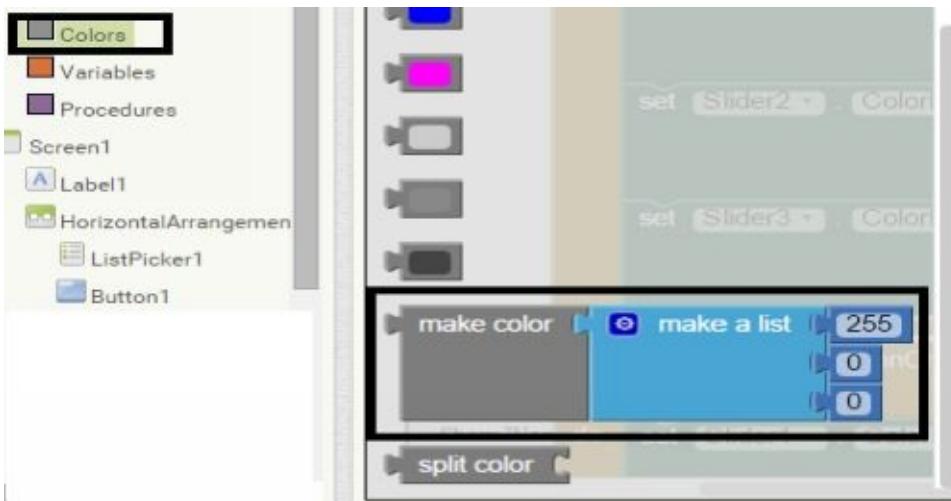
on the activity starter should remind us to turn on the Bluetooth device, and also the canvas should be set to white color. Complete the blocks given below do all this functions.



Screen initialize function can be picked from the blocks present inside the Screen1.

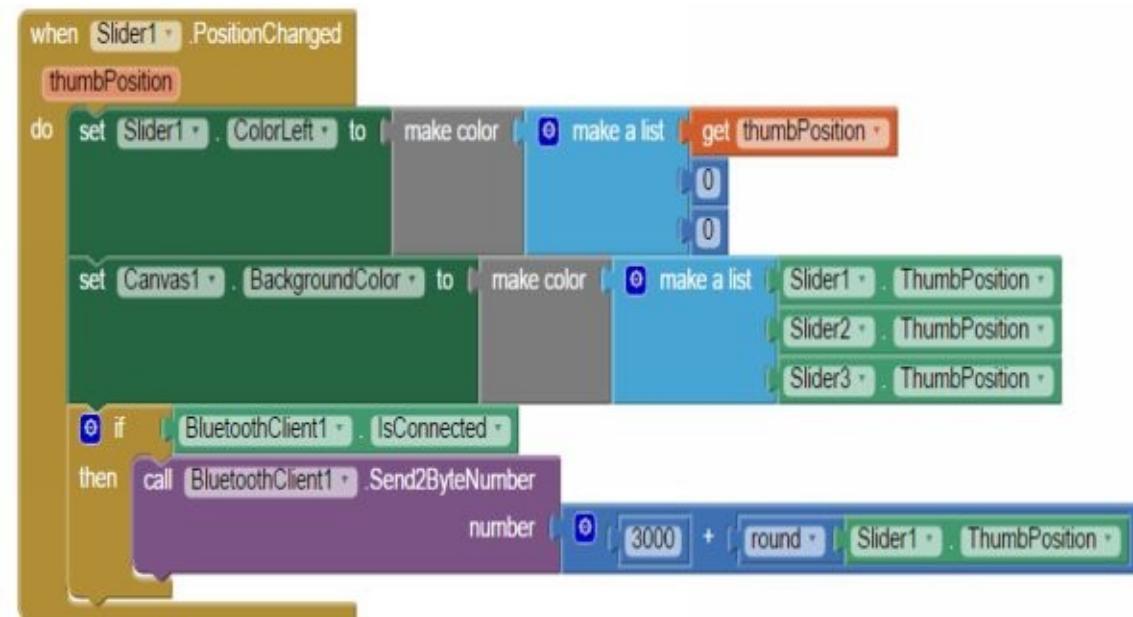
Click on activity starter and pick activity starter action and place it in screen initialize block and pick block from text and type “android.bluetooth.adapter.action.REQUEST_ENABLE” remove the quotes between the commands.

Slider color left block can be find inside the respective slider on the left side functions.



Make color block is present inside the colors built in function.

Adding two number block is presented in the math built in block, add round block before slider 1 position and also pick an empty number place from Math built in block add two numbers together.

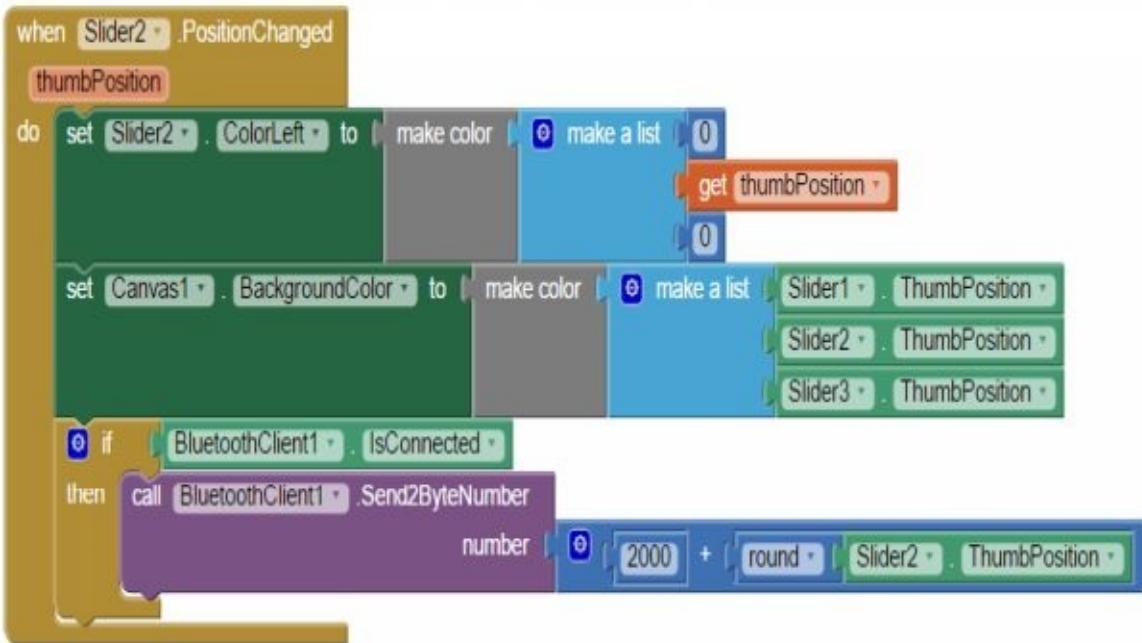


Complete above blocks to store the first value in the canvas and also when we change the first slider position if the android device is connected to Bluetooth then the slider value will be added with 3000 and send to connected Bluetooth device, we have already seen why we are sending two byte number in Multiple servo programming. Please refer if you have not completed that tutorial.

Get thumb position block can be find in the variables, pick get function in the variables and place it in the color make list and choose thumb position.

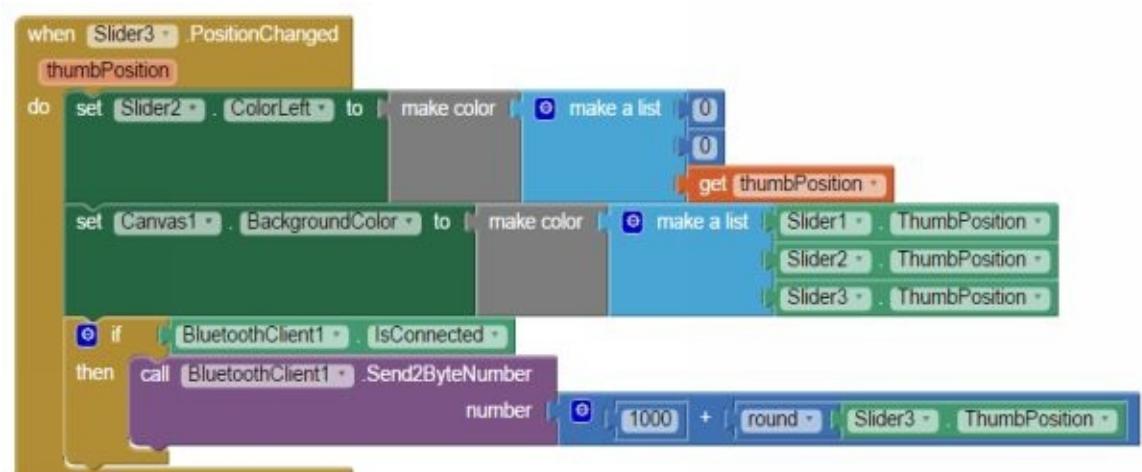


After placing get block in the make color click on the inverted triangle symbol to select thumb position. This variable will hold the current value of slider.

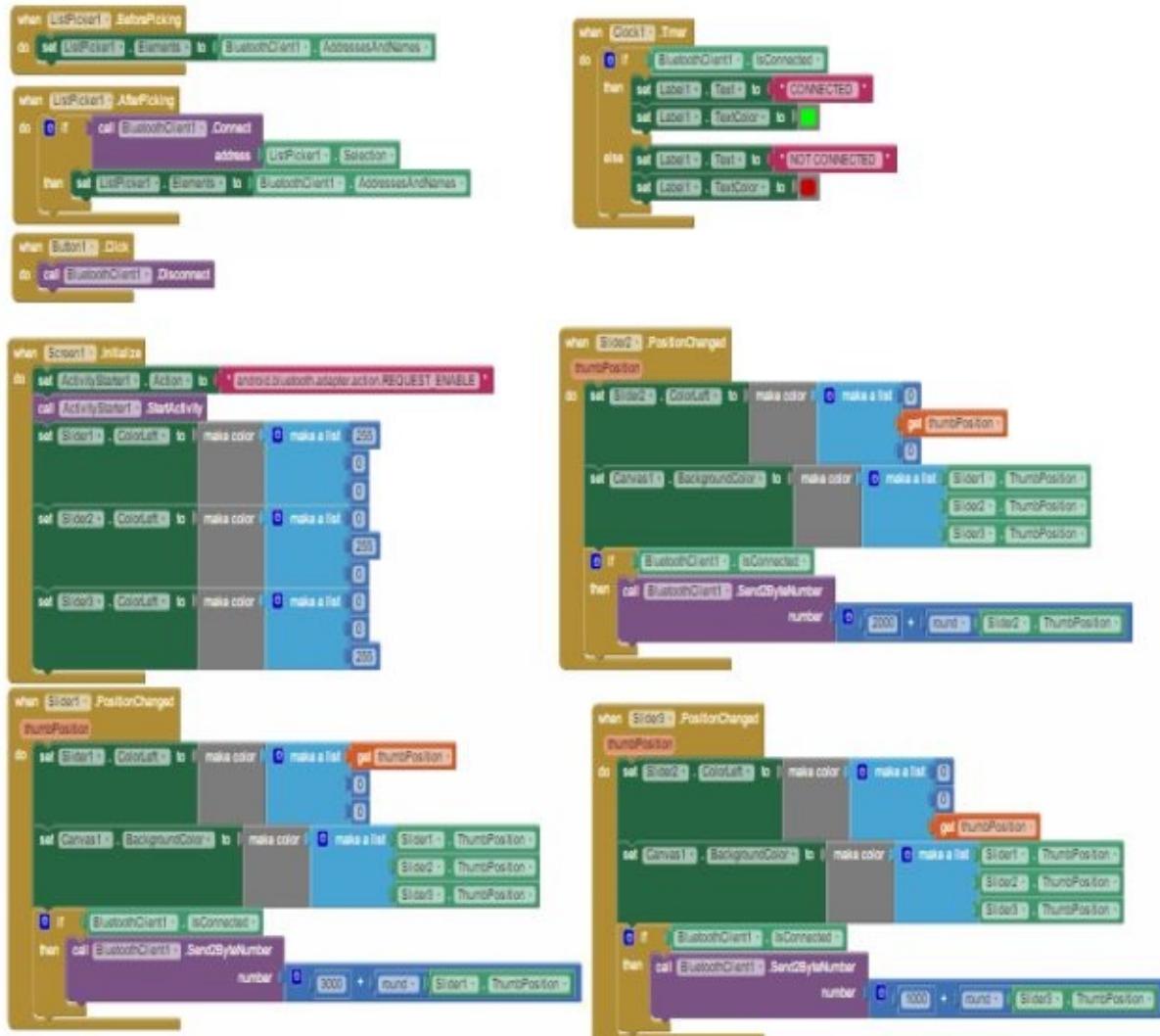


Complete the same block for the second slider, right click on the first slider and duplicate the blocks and select to slider2 wherever slider1 is presented, make sure to edit the value in the Bluetooth client 1.

Complete the third slider block to complete the program, this slider does the same function like the other two sliders.



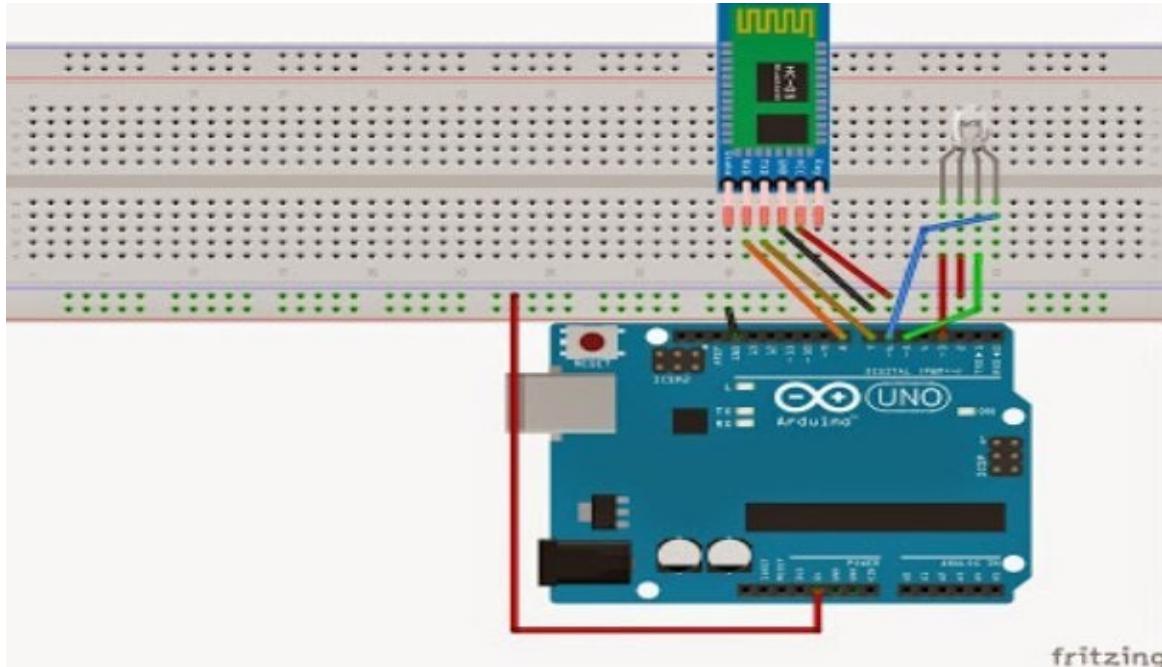
Complete program for the RGB Slider



Arduino Program for RGB Slider

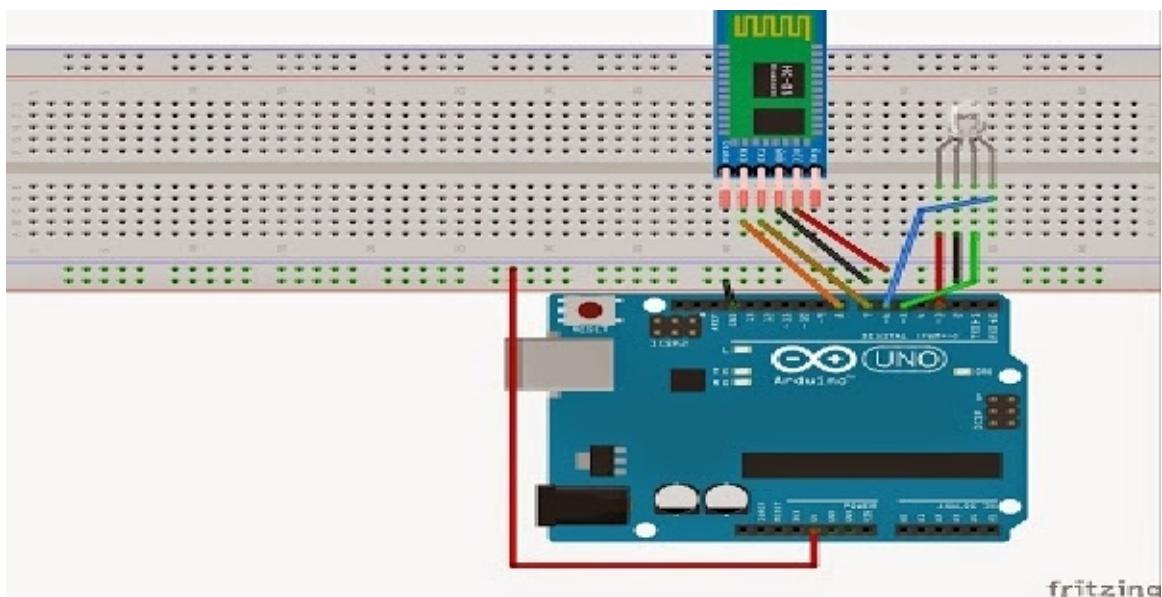
There are two types of RGB LED available in the market one is common cathode and another one is common anode. Make sure what type of RGB LED you are using for your practice.

Common Anode Type



If you have a common anode type RGB LED then you have to connect your anode to 3.3v of Arduino. Connect the red pin to 3rd pin of Arduino green pin to 5th pin of Arduino and blue pin to 6th pin of Arduino. Connect Bluetooth Tx to 7th pin and Rx to 8th pin of Arduino.

Common Cathode Type



If you have a common cathode type RGB LED then you have to connect your cathode to ground pin of Arduino. Connect the red pin to 3rd pin of Arduino green pin to 5th pin of Arduino and blue pin to 6th pin of Arduino. Connect Bluetooth Tx to 7th pin and Rx to 8th

pin of Arduino. You also need to connect 220 ohm resistor between the pins of LED and Arduino to protect it from burning.

Common anode Arduino program

```
#include <SoftwareSerial.h>

int bluetoothTx = 7;
int bluetoothRx = 8;
SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup()
{
pinMode(3,OUTPUT); // red pin of RGB LED
pinMode(5,OUTPUT); // Green pin of RGB LED
pinMode(6,OUTPUT); // blue pin of RGB LED
digitalWrite(3,HIGH);
digitalWrite(5,HIGH);
digitalWrite(6,HIGH);

//Setup usb serial connection to computer
Serial.begin(9600);

//Setup Bluetooth serial connection to android
bluetooth.begin(9600);

}

void loop()
{
//Read from bluetooth and write to usb serial
if(bluetooth.available()>= 2 )
{
unsigned int color1 = bluetooth.read();
unsigned int color2 = bluetooth.read();
unsigned int color = (color2 *256) + color1;
Serial.println(color);

if (color >= 1000 && color <1255){
int blue = color;
blue = map(blue, 1000,1255,0,255);
}
```

```
int blue1 = 255-blue;
analogWrite(6,blue1);
Serial.println(blue1);
delay(10);
}

if (color >=2000 && color <2255){
int green = color;
green = map(green,2000,2255,0,255);
int green1 = 255 - green;
analogWrite(5,green1);
Serial.println(green1);
delay(10);

}

if (color >=3000 && color < 3255){
int red = color;
red = map(red, 3000, 3255,0,255);
int red1 = 255 - red;
analogWrite(3,red1);
Serial.println(red1);
delay(10);
}

}
```

This program is same as the Mutli servo program that we did in the previous chapter.

Common Cathode Arduino program

```
#include <SoftwareSerial.h>

int bluetoothTx = 7;
int bluetoothRx = 8;

SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup()
{
pinMode(3,OUTPUT); // red pin of RGB LED
pinMode(5,OUTPUT); // Green pin of RGB LED
pinMode(6,OUTPUT); // blue pin of RGB LED
digitalWrite(3,LOW);
digitalWrite(5,LOW);
digitalWrite(6,LOW);

//Setup usb serial connection to computer
Serial.begin(9600);

//Setup Bluetooth serial connection to android
bluetooth.begin(9600);
}

void loop()
{
//Read from bluetooth and write to usb serial
if(bluetooth.available()>= 2 )
{
unsigned int color1 = bluetooth.read();
unsigned int color2 = bluetooth.read();
unsigned int color = (color2 *256) + color1;
Serial.println(color);

if (color >= 1000 && color <1255){
```

```
int blue = color;  
blue = map(blue, 1000,1255,0,255);  
analogWrite(6,blue);  
Serial.println(blue);  
delay(10);
```

```
}
```

```
if (color >=2000 && color <2255){  
int green = color;  
green = map(green,2000,2255,0,255);  
analogWrite(5,green);  
Serial.println(green);  
delay(10);
```

```
}
```

```
if (color >=3000 && color < 3255){  
int red = color;  
red = map(red, 3000, 3255,0,255);  
analogWrite(3,red);  
Serial.println(red);  
delay(10);  
}  
}
```

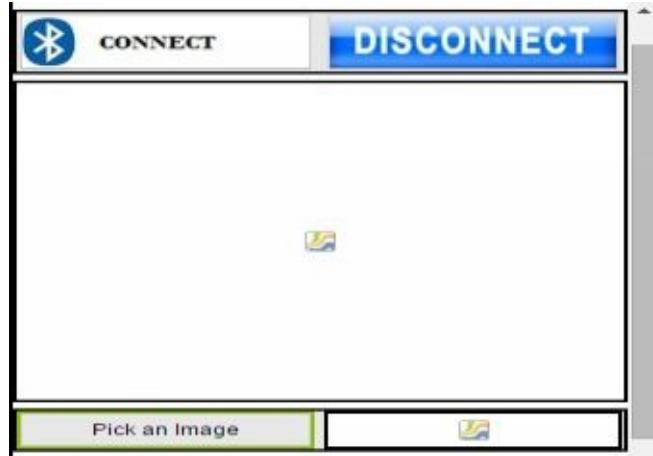
```
}
```

This program is same as the Mutli servo program that we did in the previous chapter.

RGB Color Picker

In the previous chapter we have seen how to set a color for an RGB led from android app, this app will be much similar to the previous one but instead on mixing slider we are going to choose the color's by picking existing image from gallery.

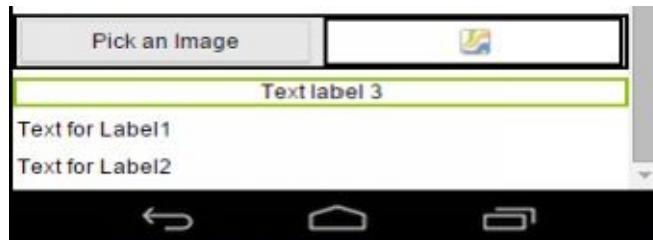
- 1) Pick horizontal arrangement and place it on the viewer screen and change its properties width to fill parent.
- 2) Pick list picker from user interface and place it on the horizontal arrangement change its image by uploading any Bluetooth image
- 3) Place a button next to list picker this button will be used for disconnecting from the Bluetooth device, upload any image to the button to make it look more attractive.
- 4) Pick canvas from Drawing and animation and place it below horizontal arrangement. Change the height of canvas to 250 pixels and width to fill parent in the canvas properties tab.
- 5) Pick horizontal arrangement and place it below canvas edit its properties and change width to fill parent.
- 6) Pick image picker from media and place it in horizontal arrangement and change its width in properties to fill parent and change the text to Pick an image.
- 7) Pick canvas from drawing and animation, place next to image picker and change its properties width to fill parent.



Place Label 1 below to image picker (this will store the canvas X data)

Place Label 2 below label 1 (this will store the canvas Y data)

Place Label 3 above Label 1 (this will store R G and B pixels) change text label 3 width to fill parent and text alignment to centre.



Place Label 4 below Text label 2(this label will store RED pixel value)

Place Label 5 below Text label 4(this label will store GREEN pixel value)

Place Label 6 below Text label 5(this label will store BLUE pixel value)



We usually add a label to display to know connectivity between Bluetooth devices, a label need to add for this purpose. Add label 7 at the top of Bluetooth and disconnect



Place Image from user Interface, and upload any RGB image to it and uncheck the visible box in the properties of Image.



Finally place non visible components for this app, Bluetooth client and Activity starter from connectivity and clock from sensors.

Designer for Viewer screen has completed and it's time to program for the viewer screen, click on the blocks to program it.

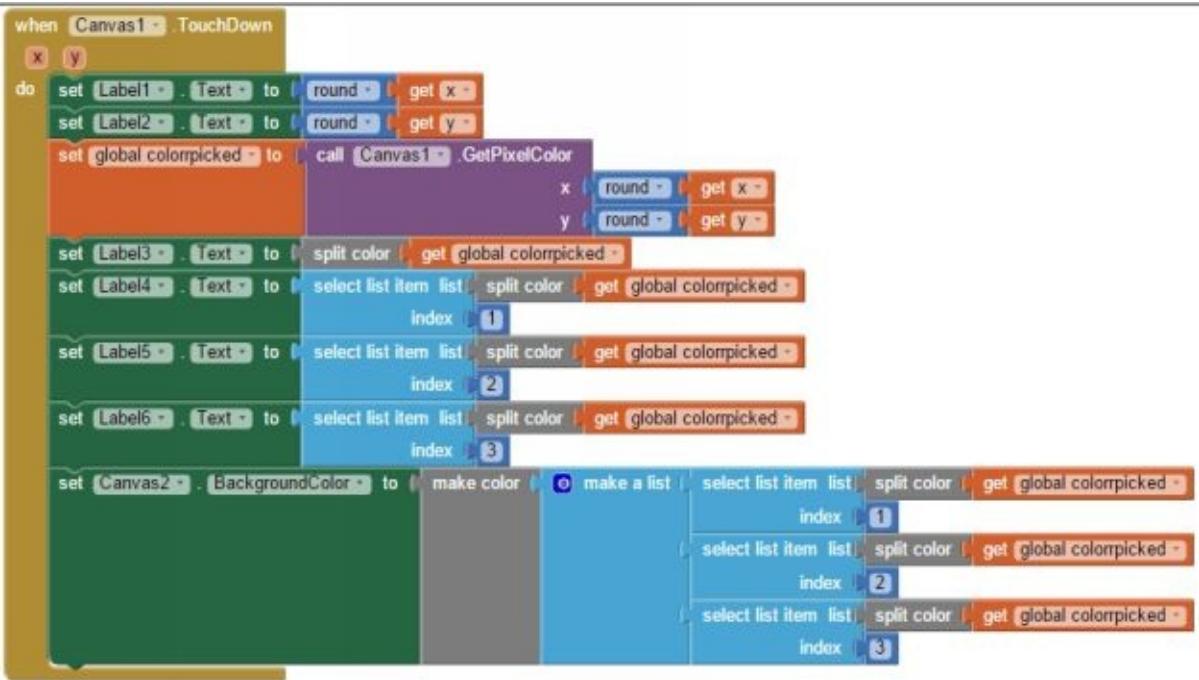
Initialize two global variables, this can be of any value and I choose “49” and “a” you can also choose any values.

```
initialize global colorpicked to "49"
initialize global previouscolor to "a"
```

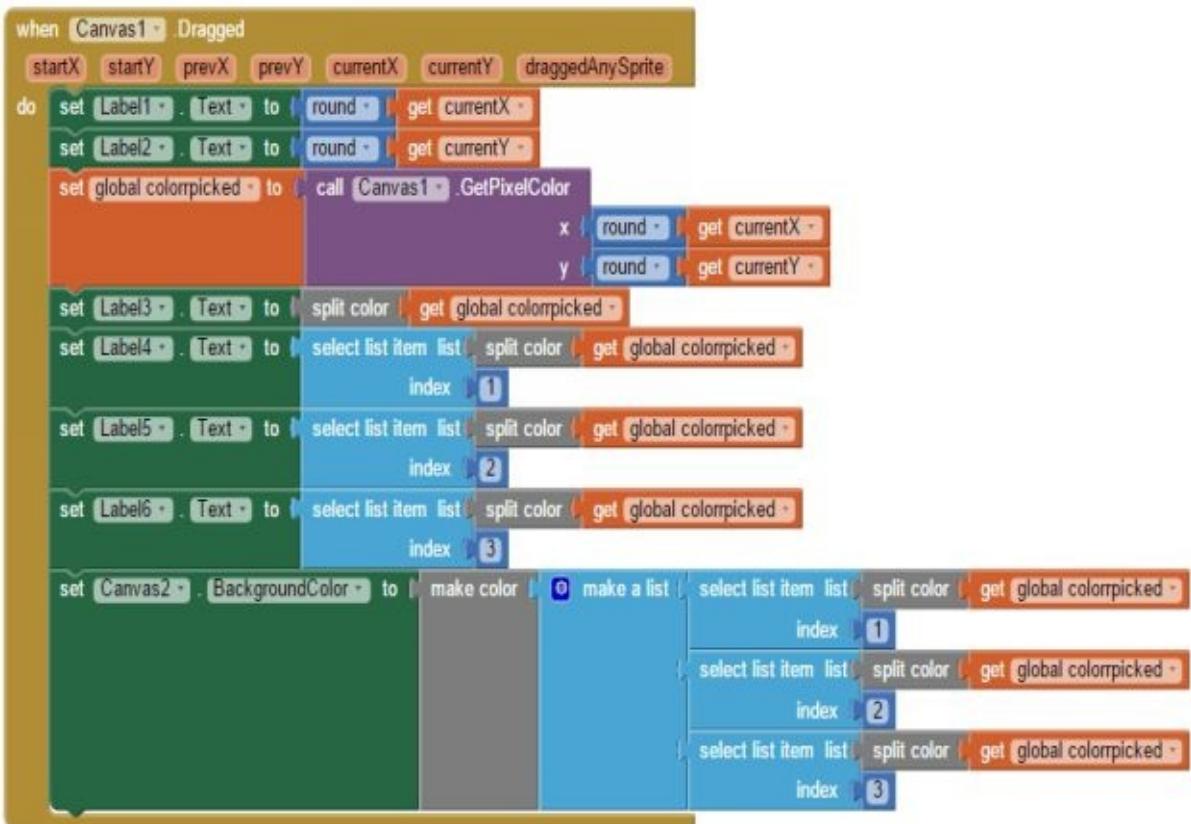
Initialize global blocks can be find the variables and change the names to colorpicked and previouscolor.

```
when Screen1.Initialize
do
  set ActivityStarter1 Action to " android.bluetooth.adapter.action.REQUEST_ENABLE "
  call ActivityStarter1 StartActivity
  set global colorpicked to call Canvas1 GetPixelColor
    x round 1
    y round 1
  set global previouscolor to get global colorpicked
  set Canvas1 BackgroundImage to Image1 Picture
```

Complete the above blocks, this will initialize the screen and start the activity starter will ask us to enable the Bluetooth function if it's turned off, default image will be displayed in the canvas.



Above blocks will split RGB color and store it in label 4, 5, 6, when you touch certain part of the image the above block help to identify RGB values and the values will be displayed in the label 3.



When you drag around the image, the above blocks will calculate the RGB pixel values and separate these colors and store it in the respective labels.

```

when ImagePicker1 .AfterPicking
do set Canvas1 .BackgroundImage to ImagePicker1 .Selection

when Button1 .Click
do call BluetoothClient1 .Disconnect

when ListPicker1 .BeforePicking
do set ListPicker1 .Elements to BluetoothClient1 .AddressesAndNames

when ListPicker1 .AfterPicking
do if call BluetoothClient1 .Connect
    address ListPicker1 .Selection
then set ListPicker1 .Elements to BluetoothClient1 .AddressesAndNames

```

Above block will set an image to the canvas that is currently picked from the gallery, when you click the button it will disconnect from the Bluetooth device that has currently connect to.

```

when Clock1 .Timer
do if BluetoothClient1 .IsConnected
then set Label7 .Text to "connected"
set Label7 .TextColor to green
call BluetoothClient1 .Send2ByteNumber
    number 1000 + Label4 .Text
call BluetoothClient1 .Send2ByteNumber
    number 2000 + Label5 .Text
call BluetoothClient1 .Send2ByteNumber
    number 3000 + Label6 .Text
else set Label7 .Text to "Not connected"
set Label7 .TextColor to red

```

This block will set the label 7 to connect or disconnect based upon the app connected to Bluetooth devices. When this app connected to the Bluetooth devices, this will send the RGB values from the Label 4, 5 and 6. **Complete code as follows**

```

initialize global [colorpicked] to [49]
initialize global [previouscolor] to [8]

when [Screen1] .Initialize
do
  set [ActivityStarter1] . Action to [android.bluetooth.adapter.action.REQUEST_ENABLE]
  call [ActivityStarter1] . StartActivity
  set [global colorpicked] to [call [Canvas1] . GetPixelColor
    x [round] [1]
    y [round] [1]]
  set [global previouscolor] to [get [global colorpicked] ]
  set [Canvas1] . BackgroundImage to [Image1] . Picture

```

```

when [ImagePicker1] .AfterPicking
do set [Canvas1] . BackgroundImage to [ImagePicker1] . Selection

```

```

when [Button1] .Click
do call [BluetoothClient1] . Disconnect

```

```

when [ListPicker1] .BeforePicking
do set [ListPicker1] . Elements to [BluetoothClient1] . AddressesAndNames

```

```

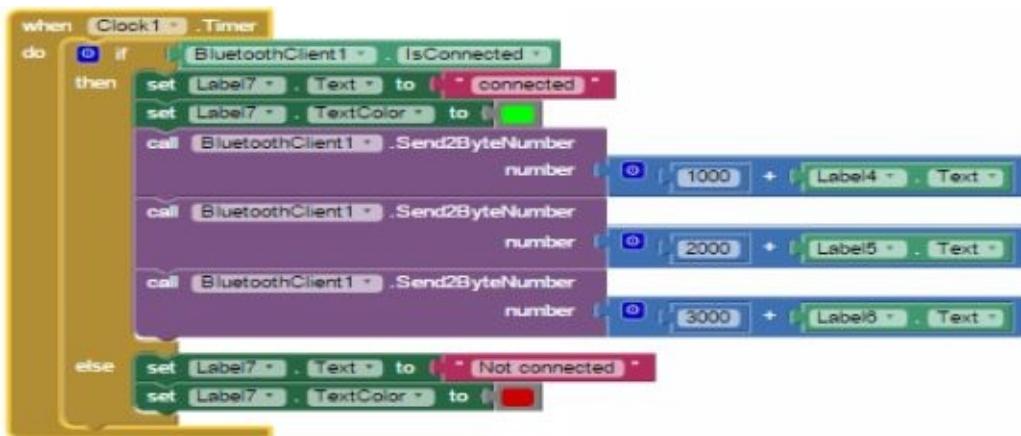
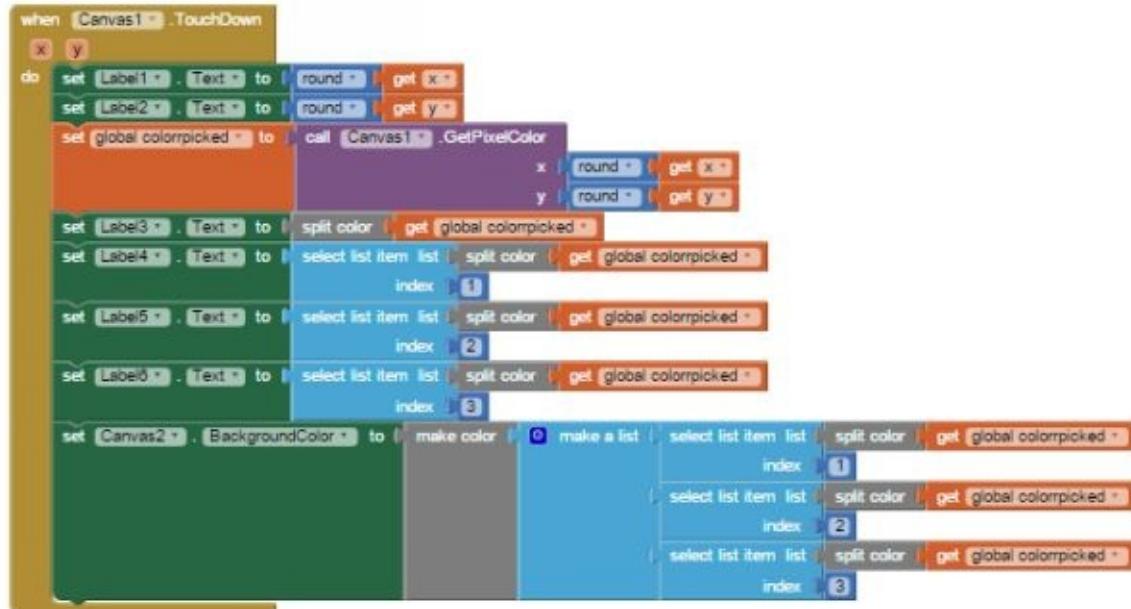
when [ListPicker1] .AfterPicking
do
  if [call [BluetoothClient1] . Connect
    address [ListPicker1] . Selection]
  then set [ListPicker1] . Elements to [BluetoothClient1] . AddressesAndNames

```

```

when [Canvas1] .Dragged
startX [startY] prevX [prevY] currentX [currentY] draggedAnySprite
do
  set [Label1] . Text to [round] [get [currentX] ]
  set [Label2] . Text to [round] [get [currentY] ]
  set [global colorpicked] to [call [Canvas1] . GetPixelColor
    x [round] [get [currentX] ]
    y [round] [get [currentY] ]]
  set [Label3] . Text to [split color] [get [global colorpicked] ]
  set [Label4] . Text to [select list item list [split color] [get [global colorpicked] ]
    index [1] ]
  set [Label5] . Text to [select list item list [split color] [get [global colorpicked] ]
    index [2] ]
  set [Label6] . Text to [select list item list [split color] [get [global colorpicked] ]
    index [3] ]
  set [Canvas2] . BackgroundColor to [make color [0] make a list
    select list item list [split color] [get [global colorpicked] ]
    index [1]
    select list item list [split color] [get [global colorpicked] ]
    index [2]
    select list item list [split color] [get [global colorpicked] ]
    index [3] ]

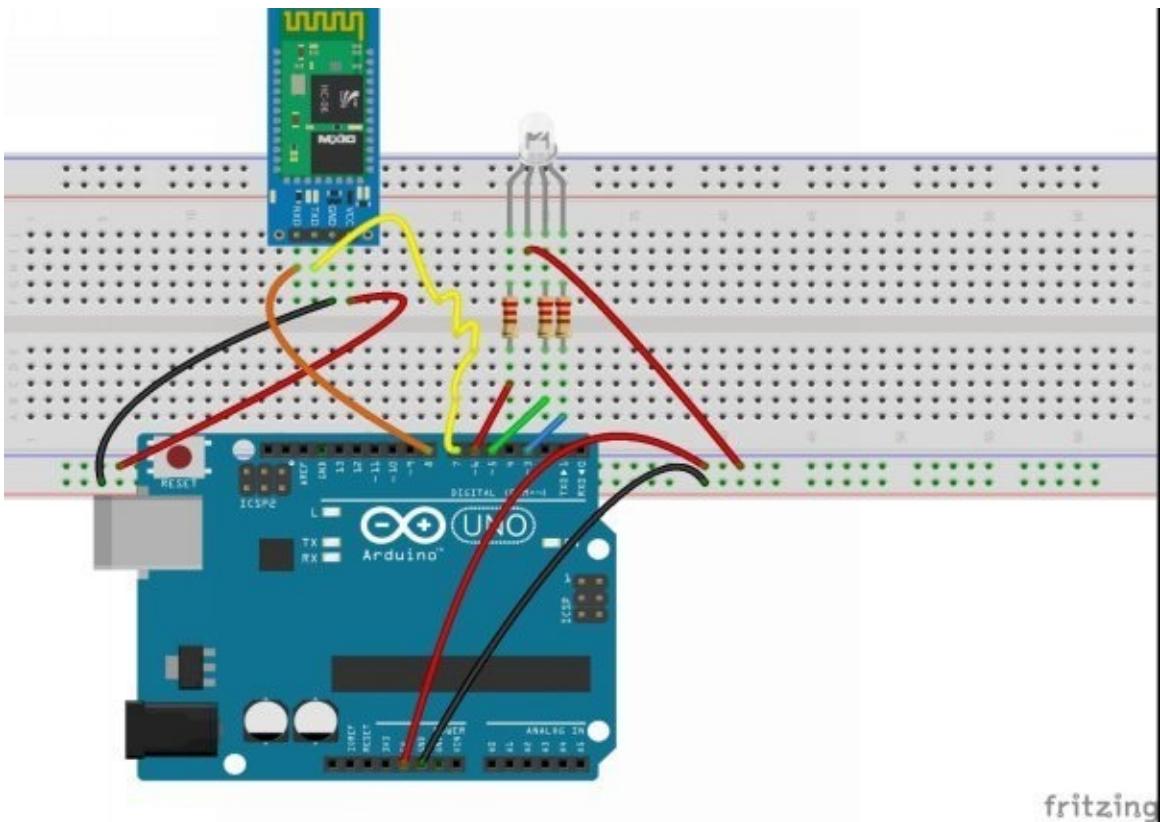
```



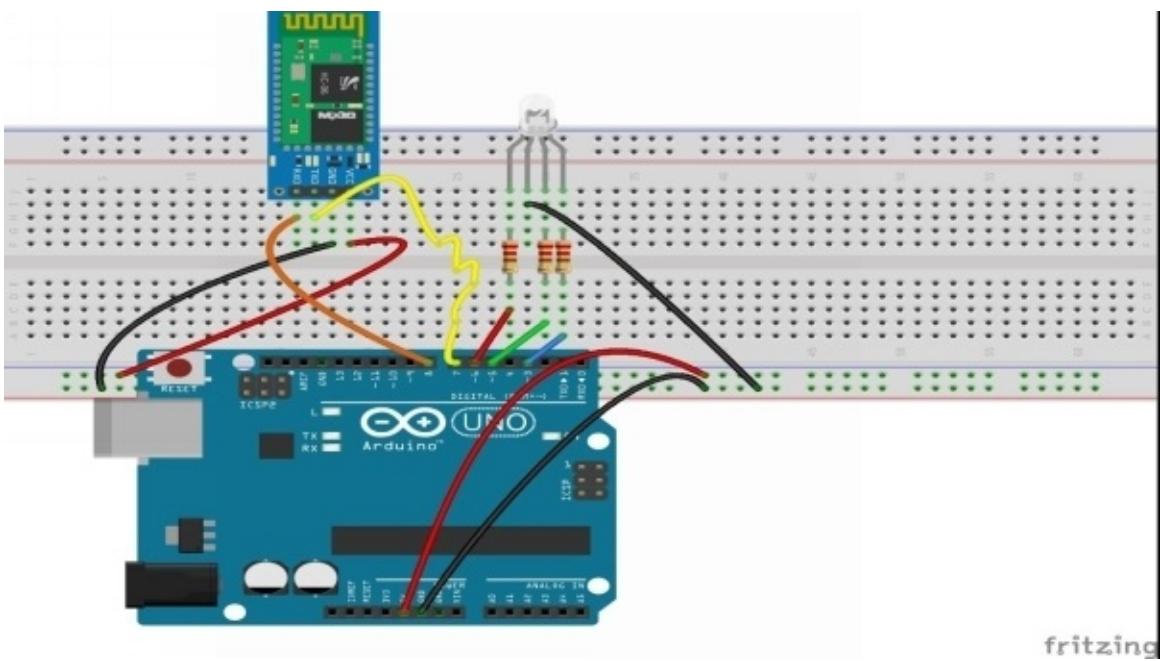
Arduino Program

This RGB color picker uses same code we used in the RGB color slider, but in this chapter you have to change the 3rd pin to connect RGB blue pin and 6th pin for RED and 5th pin for Green. Copy and paste the same Arduino code that we used for RGB color slider.

Circuit connection for common anode:



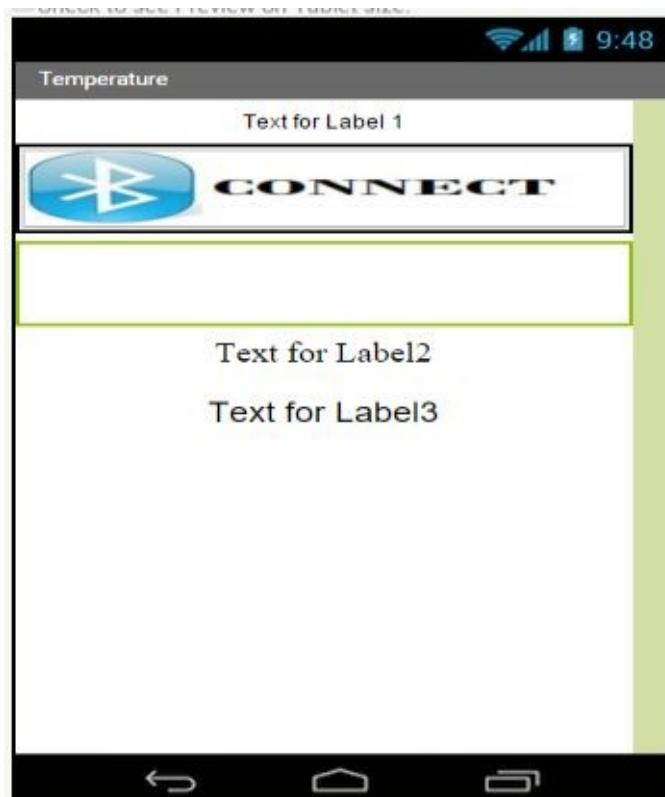
Circuit connection for common cathode:



Temperature data into Android

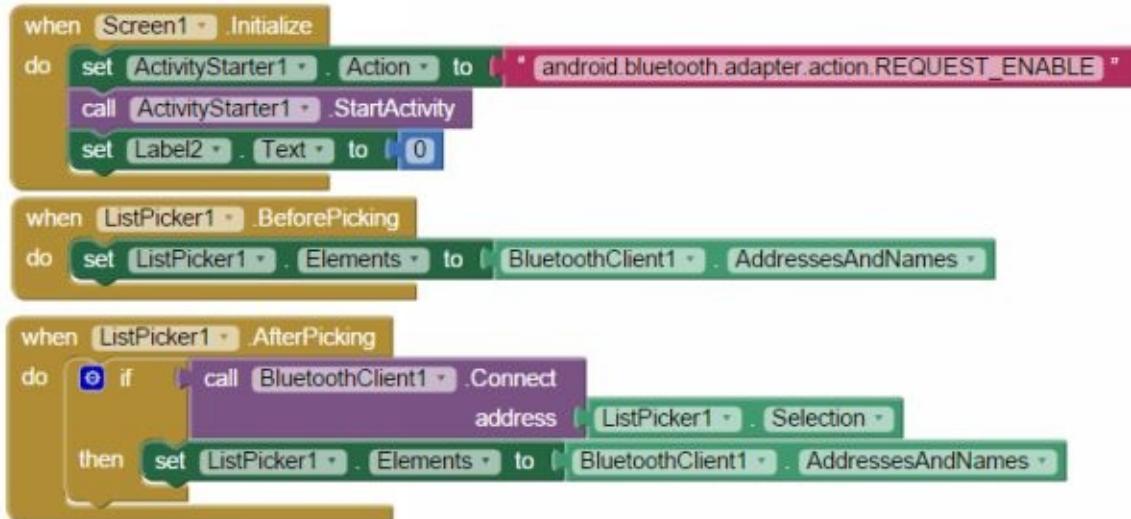
In this chapter we going to get temperature data from Arduino and display the temperature in android app, and we also going set certain conditions if the temperature is below the set temperature then in our android app it will display as cold, if the temperature exceeded the set temperature then it will display as Hot and if it falls into our set temperature then it will display text as moderate, we also going to change the background picture based upon these temperatures.

- 1) Pick label-1 and place it on top of viewer screen.
- 2) Pick horizontal arrangement place it below label-1 and change its properties as we normally do
- 3) Place list picker from user interface into horizontal arrangement change its name to Bluetooth and width to fill parent
- 4) Place horizontal arrangement below list picker, we use this for empty space between list picker and a label to follow.
- 5) Pick label and place it below empty horizontal arrangement
- 6) Pick another label and place it below the previous label.
- 7) Pick image from user interface and place it on viewer screen upload an image for cold condition and uncheck the visible in the properties to hide the image, pick and place another image and upload an image for moderate condition, do the same for Hot condition.

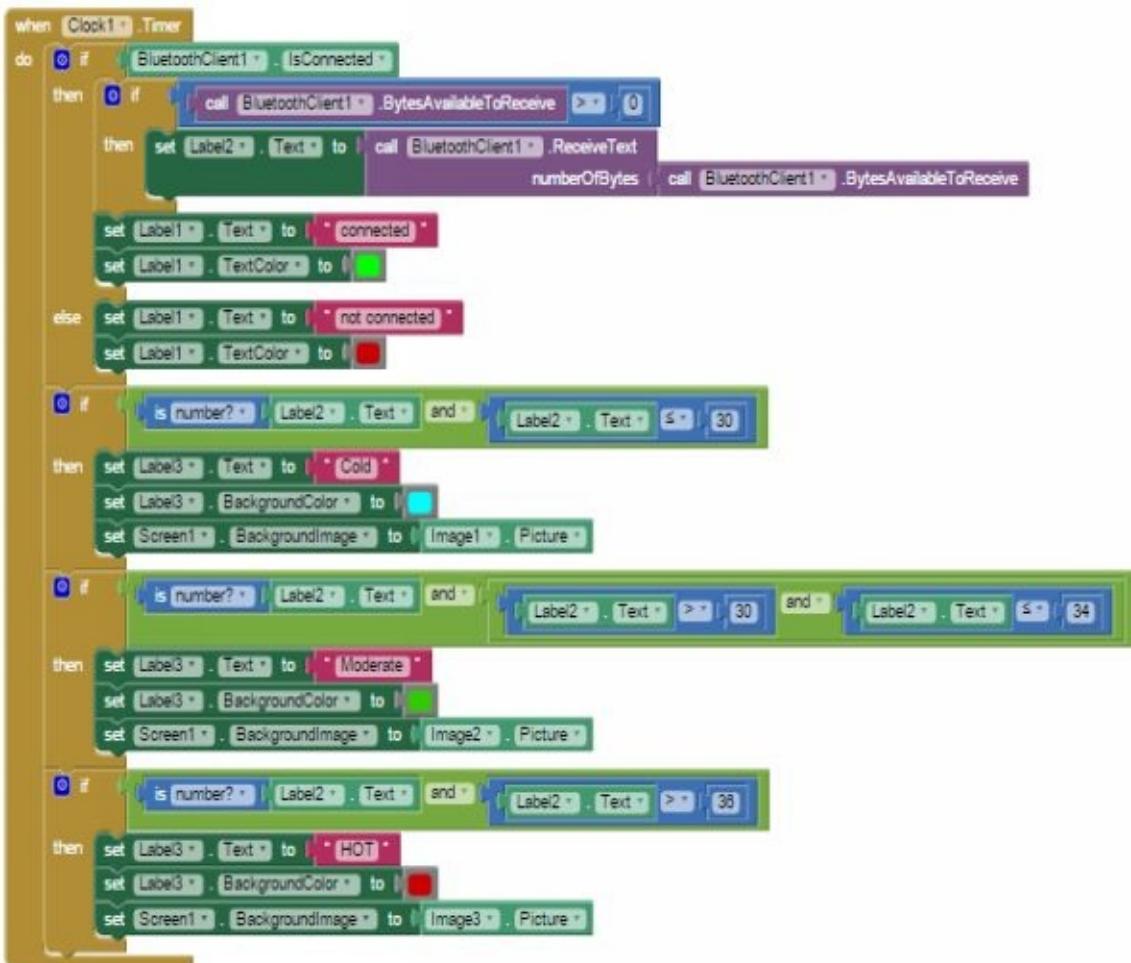


Place non visible components Bluetooth Client, Clock and Activity starter, activity starter can be placed from the connectivity palette.

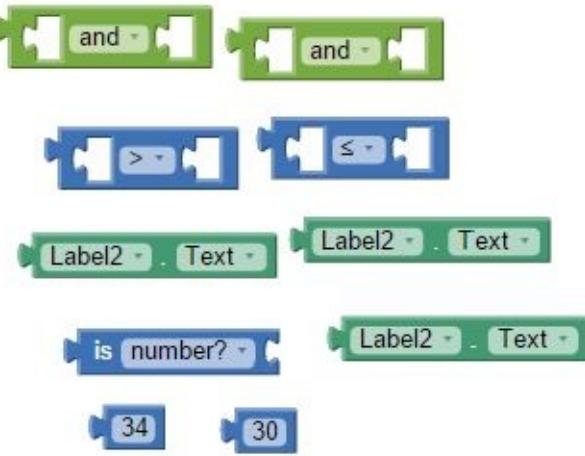
Move to blocks for programming for the Designer we have completed, click on the blocks and start to program the below blocks.



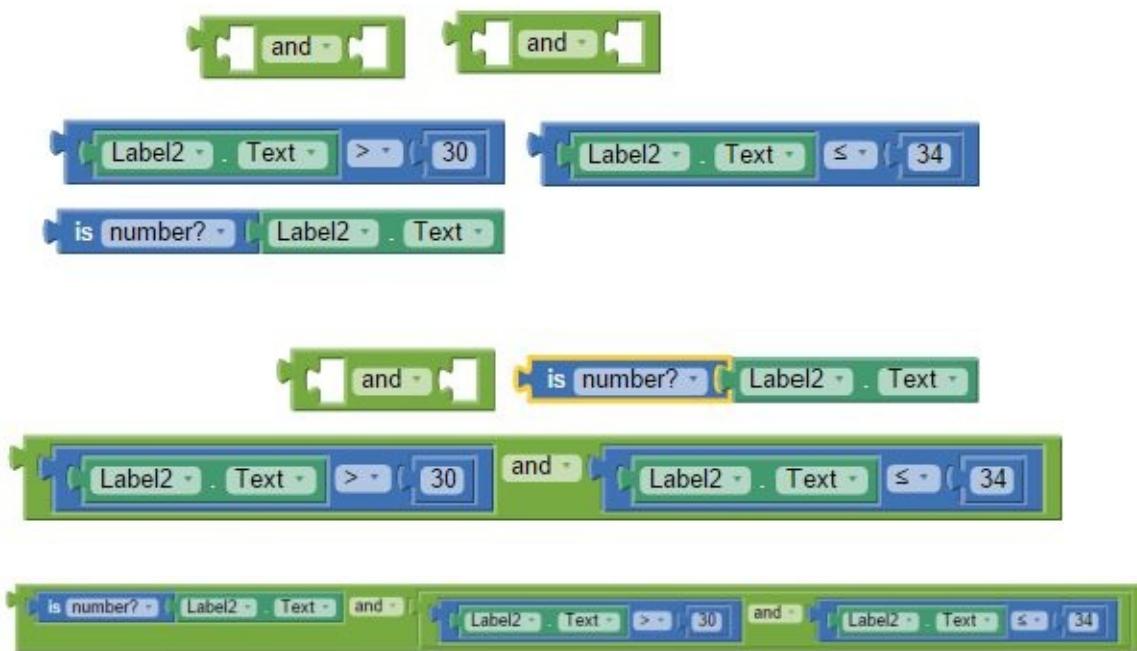
The above blocks will connect to the Bluetooth device, and when app is opened for the first time and if the Bluetooth is disabled activity starter will remind us to turn on the Bluetooth. Complete the below code to finish the program.



Blocks used in the If condition



And blocks can be found in the logic of inbuilt function, less than and greater or equal to block, is number? Will be presented inside the inbuilt math function.

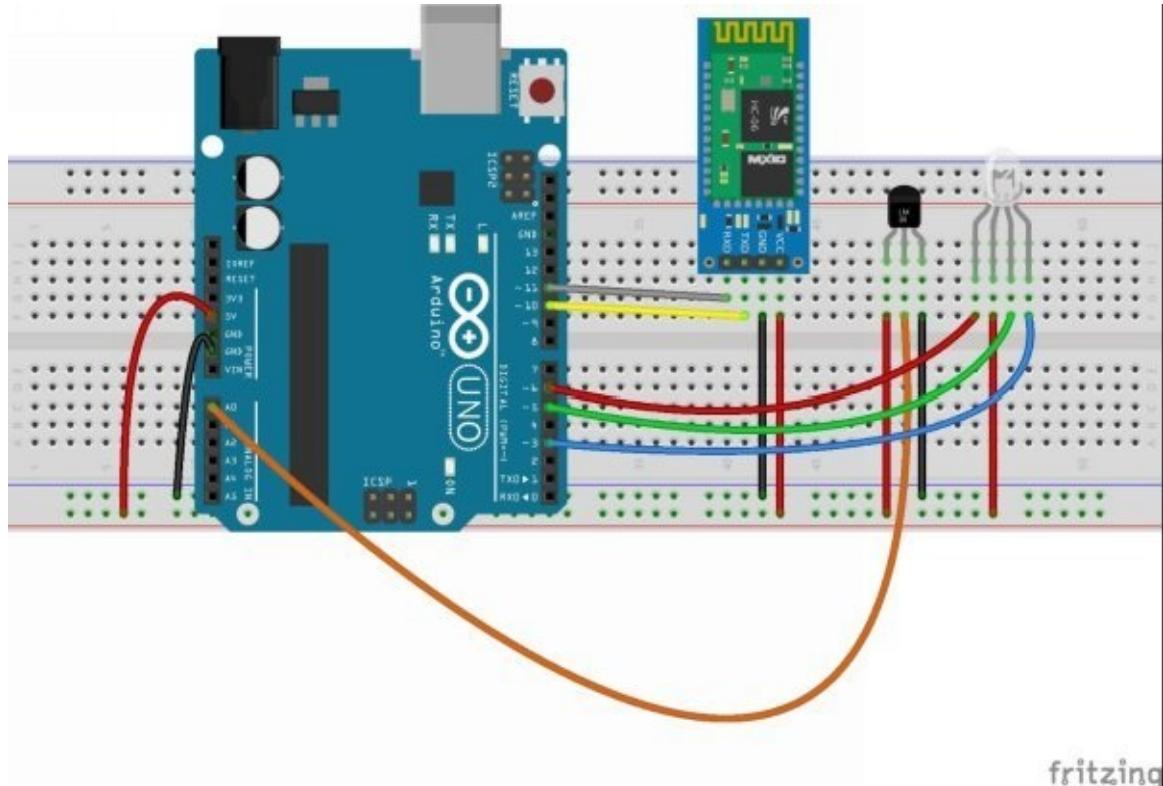


This is exactly like playing a puzzle where you have to place a block inside another to make it work.

The conditions are used to check whether the temperature exceed or below the set condition and it also set the image to the background based on the temperature.

Arduino Program for Temperature

On the Arduino side for this chapter we going to use LM35 temperature sensor, or any other temperature sensor you would like to use can be used for this chapter, if you are changing temperature sensor make sure you also changing the code based on the data sheet for the sensor.



In the above circuit connection, Bluetooth, LM35 temperature sensor and common anode type RGB LED are used.

Connect Bluetooth Tx pin to 10th pin of Arduino and Rx to 11th pin of Arduino.

Connect output pin LM35 to A0 pin of Arduino.

Connect red pin of RGB LED to 6th pin of Arduino.

Connect green pin of RGB LED to 5th pin of Arduino.

Connect blue pin of RGB LED to 3rd pin of Arduino.

Arduino Program:

```
// code begins here:
```

```

#include<SoftwareSerial.h>

const int Tempsensor = A0; // connect output pin of lm35 to A0 pin of arduino
int bluetoothTx = 10; // connect tx of bluetooth to 10th pin
int bluetoothRx = 11; // connect rx of bluetooth to 11th pin

SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);

void setup() {
    pinMode(Tempsensor, INPUT);
    pinMode(3,OUTPUT);//blue pin of rgb led
    pinMode(4,OUTPUT);// green pin of rgb led
    pinMode(5,OUTPUT);//red pin of rgb led
    Serial.begin(9600);// serial communication begins
    bluetooth.begin(9600);// bluetooth communication begins
}

void loop() {
    float temp = analogRead(Tempsensor); // read temperature sensor voltage
    temp = temp * 0.48828125; // (5*1000)/1024 = 0.48828125 from sensor data sheet
    Serial.println(" ");
    Serial.println(temp); // watch the current temperature in serial monitor
    bluetooth.print(" ");
    bluetooth.println(temp); // send temperature to bluetooth device
    delay(200);

    if (temp <= 30){ // if temperature is below 30 turn on blue light
        digitalWrite(3,LOW);
        digitalWrite(4,HIGH);
        digitalWrite(5,HIGH);
    }

    else if (temp >30 && temp <=34){ // if temp is above 30 and below 34 turn on green
light
}

```

```
digitalWrite(3,HIGH);
digitalWrite(4,LOW);
digitalWrite(5,HIGH);

}

else if(temp >36){ // if temperature is above 36 turn on red light
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
    digitalWrite(5,LOW);

}

// code ends here
```

Type the above program in Arduino ide and upload to Arduino board, sensor data from LM35 are calculated and send from Arduino to android device, android device receive these data through app we created in the previous step, temperature data will be checked with the set conditions

Condition 1: if the temperature is less than 30, upload background image to photo related to cold condition and change the label 3 to cold

Condition 2: if the temperature is greater than 30 and less than 34, upload background image to photo related to moderate condition and change the label 3 to moderate

Condition 3: if the temperature exceeds 34 then set the image to hot condition and also change the label to HOT.

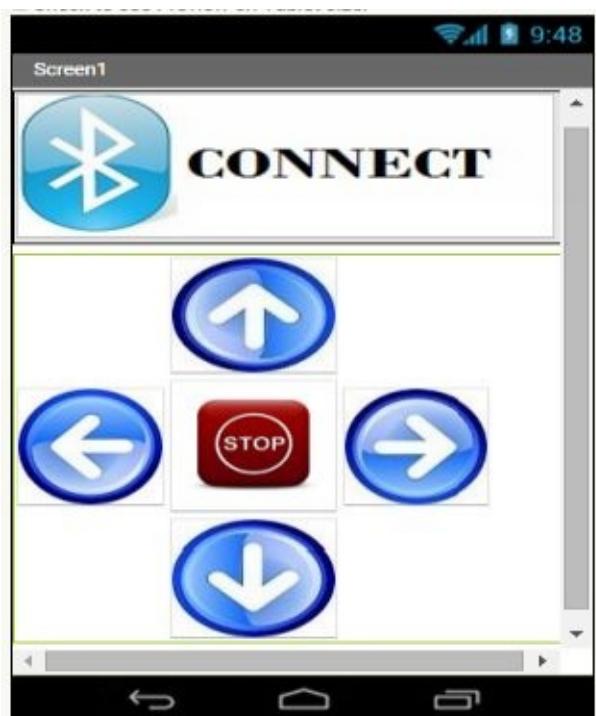
Android Remote control Robot

This app is very similar to the first chapter we created to turn on and off a single LED that has connected to Arduino, in this chapter we going to send 5 different commands from android app and execute certain functions on the Arduino side.

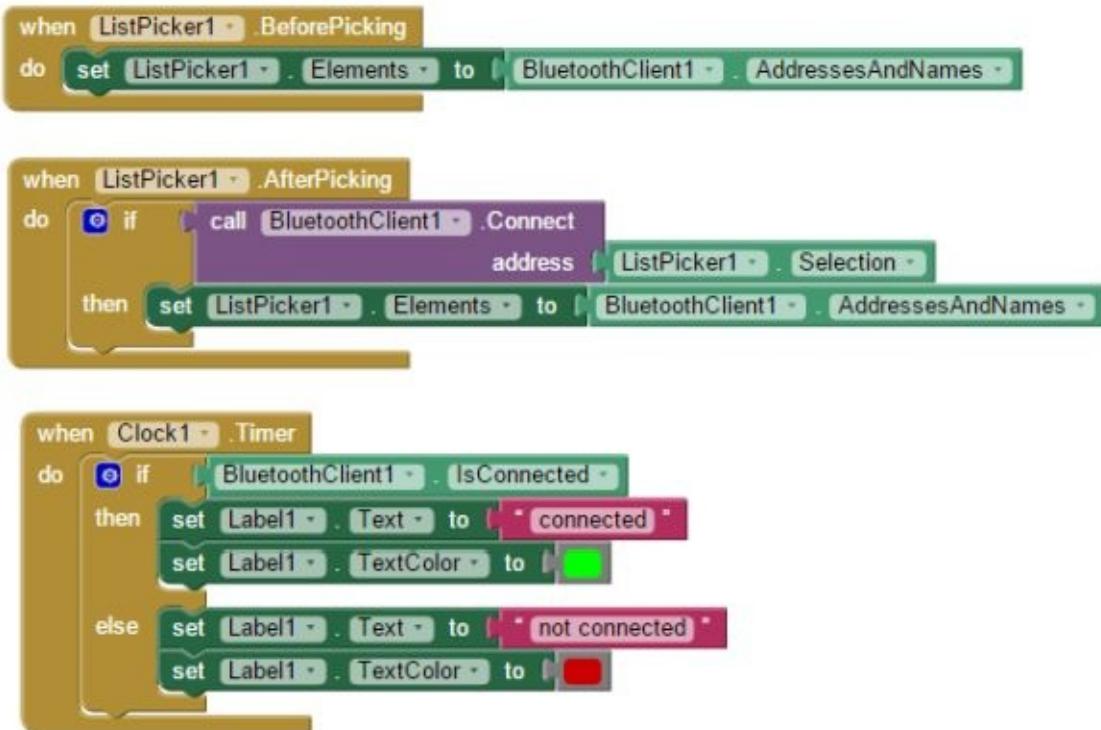
- 1) Pick a label and place it on the top of Viewer Screen and edit the properties of the label
- 2) Pick horizontal arrangement and place it below label.
- 3) Pick list picker and place it inside horizontal arrangement and change the background image of list picker to some Bluetooth image.
- 4) Pick table arrangement from layout edit the properties of table arrangement and enter column as 3 and row as 3 and width to fill parent.



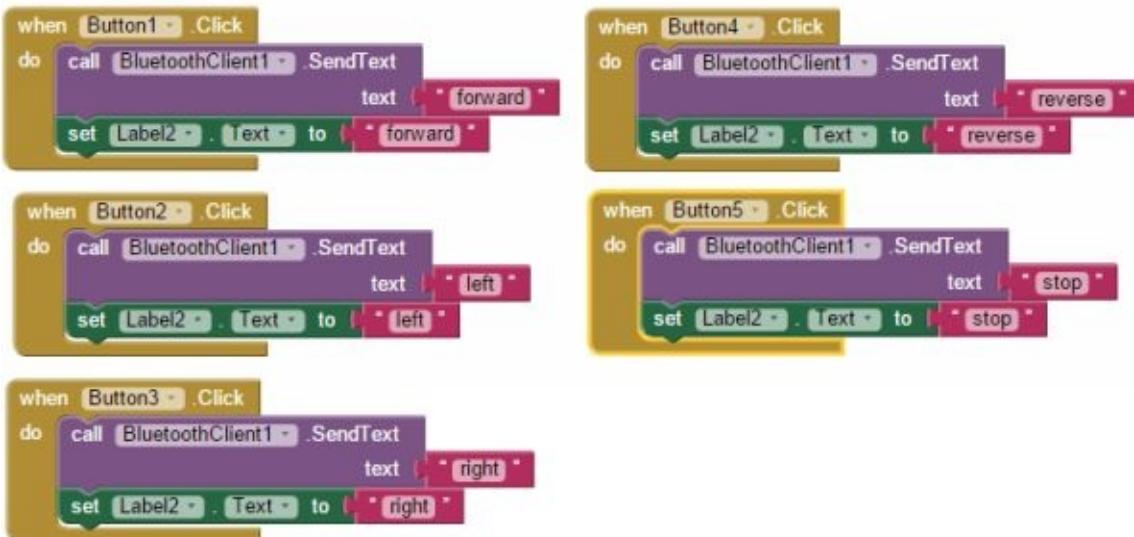
- 5) Pick button from user interface and place it around the table, arrange five button and change the images of the button to reflect like the below screenshot.



After completing the above Viewer screen move on to blocks by clicking on the blocks on the upper right side in the MIT app inventor window.



Complete the above block for Bluetooth communication and also notification in the text label if the android device connected to Bluetooth device connected to Arduino.



Complete the above blocks to finish the program, when you press the up arrow that button is associated to button1 it should send forward as command, when you press down arrow that button is associated to button4 it should send reverse, likewise the command will send from the app based on the selection of button. Think about the button connectivity when programming that has not detailed in this chapter as we crossed many tutorials this is really simple think yourself and accomplish the task.

Arduino Program for Android remote

For making this project you need many components, in the previous chapters we have used simple components and we never used more than 2 components in all the chapters, when coming to robot, it involves lot of components to make it work

Follows the list of components:

Arduino Uno

Bluetooth Module (HC-05 or HC-06)

L293D Motor driver

2x Gear Motors

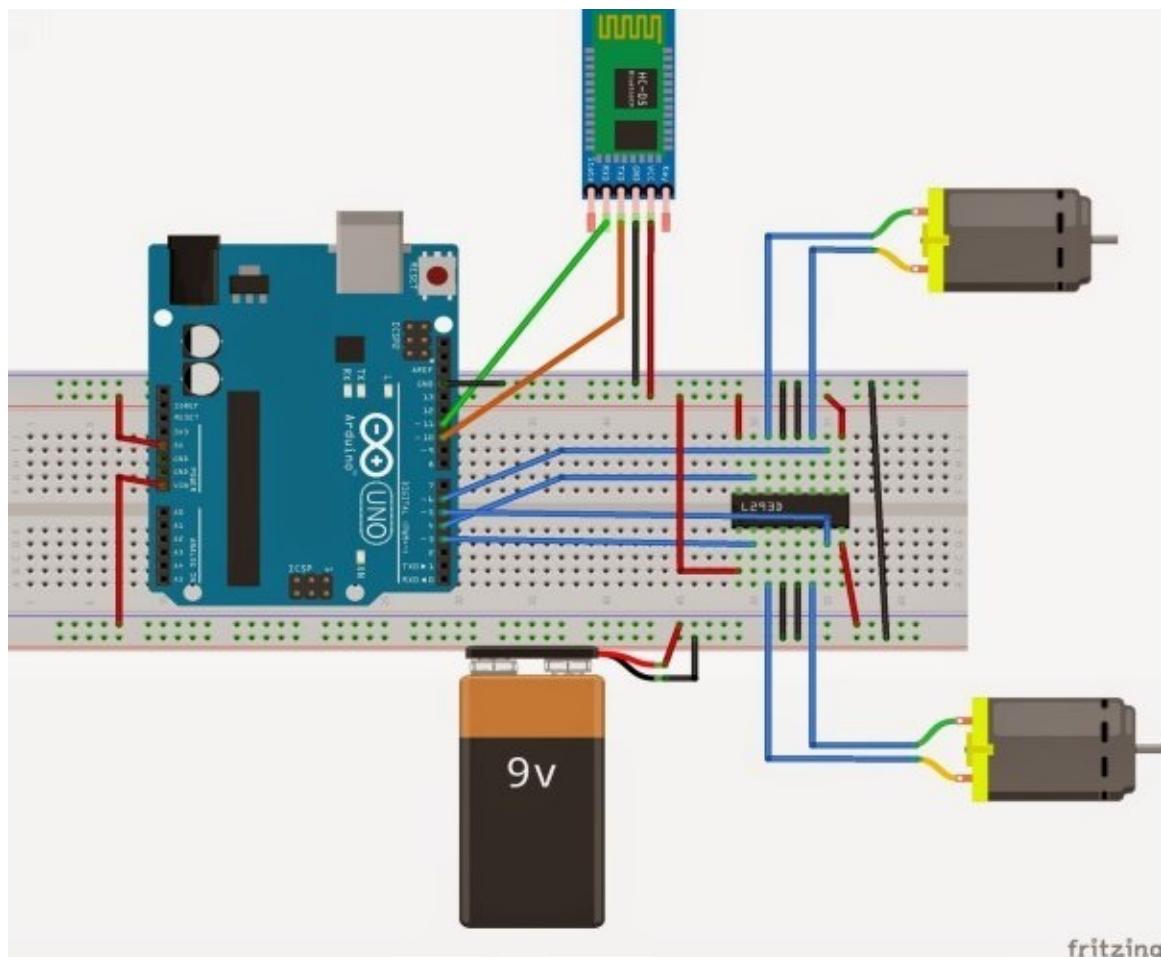
1x castor wheel

2x Wheels for Gear motor

9v battery

Robot chassis

1x breadboard and connecting wires.

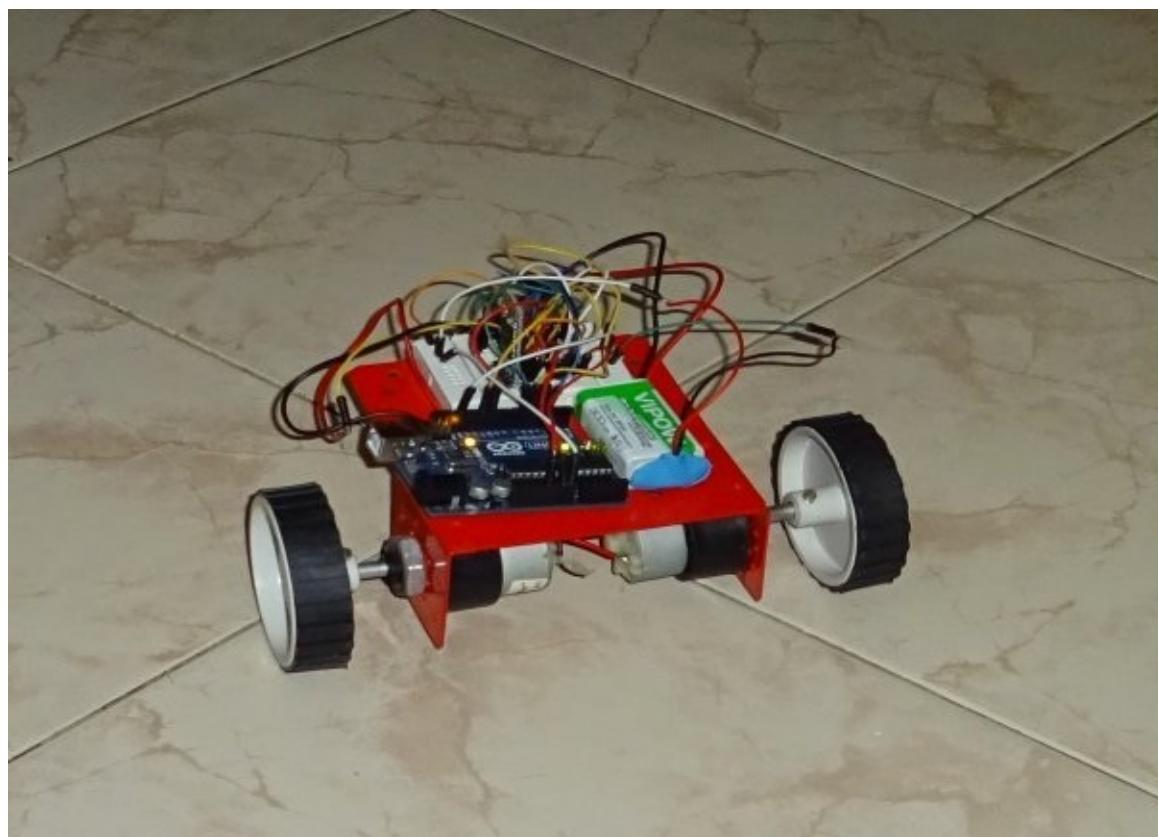


Connect the components as shown in the circuit, it is bit complicated if you are starting with electronics and robotics, instead of L293d IC you use any Motor driver module which eliminate fuzzy connection to the IC.

This robot works based on the signal passed from Arduino to the motor driver IC, all the motor driver have some logic behind it.

Above images shows the motor driver pin configuration and logic behind the motor driver, when you supply High and Low signal to the input 1 and input 2 Motor connected to Output1 and Output2 will rotate clockwise, when the signal is changed to Low and High motor will start to rotate in the anticlockwise.

Using the same logic, when we receive a forward command from android app, we have to send a High to input 1 and low to input 2 and High to input 4 and low to input 3 this make two motors to rotate forward.



Arduino code:

```
// code begins here  
#include <SoftwareSerial.h>
```

```
SoftwareSerial BT(10, 11); //TX, RX respectively
String readdata;

void setup() {
    BT.begin(9600);
    Serial.begin(9600);
    pinMode(3, OUTPUT); // connect to input 1 of l293d
    pinMode(4, OUTPUT); // connect to input 4 of l293d
    pinMode(5, OUTPUT); // connect to input 3 of l293d
    pinMode(6, OUTPUT); // connect to input 2 of l293d

}

//—————//


void loop() {
    while (BT.available()){ //Check if there is an available byte to read
        delay(10); //Delay added to make thing stable
        char c = BT.read(); //Conduct a serial read
        readdata += c; //build the string- “forward”, “reverse”, “left” and “right”
    }
    if (readdata.length() > 0) {
        Serial.println(readdata); // print data to serial monitor
        // if data received as forward move robot forward
        if(readdata == “forward”)
        {
            digitalWrite(3, HIGH);
            digitalWrite (4, HIGH);
            digitalWrite(5,LOW);
            digitalWrite(6,LOW);
            delay(100);
        }
        // if data received as reverse move robot reverse
    }
}
```

```
else if(readdata == "reverse")
{
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);
    digitalWrite(6,HIGH);
    delay(100);
}

// if data received as right turn robot to right direction.

else if (readdata == "right")
{
    digitalWrite (3,HIGH);
    digitalWrite (4,LOW);
    digitalWrite (5,LOW);
    digitalWrite (6,LOW);
    delay (100);

}

// if data received as left turn robot to left direction

else if ( readdata == "left")
{
    digitalWrite (3, LOW);
    digitalWrite (4, HIGH);
    digitalWrite (5, LOW);
    digitalWrite (6, LOW);
    delay (100);
}

// if data received as stop, halt the robot

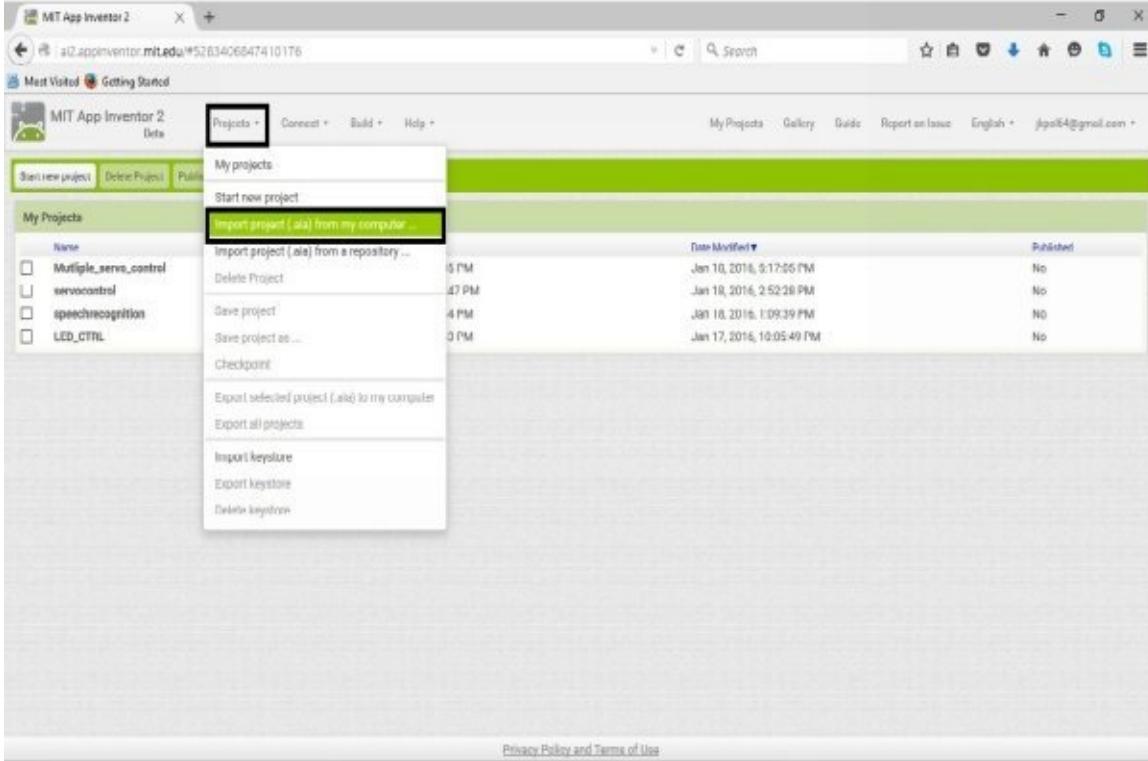
else if (readdata == "stop")
```

```
{  
    digitalWrite (3, LOW);  
    digitalWrite (4, LOW);  
    digitalWrite (5, LOW);  
    digitalWrite (6, LOW);  
    delay (100);  
}  
readdata=""}; } //Reset the variable  
// code ends here
```

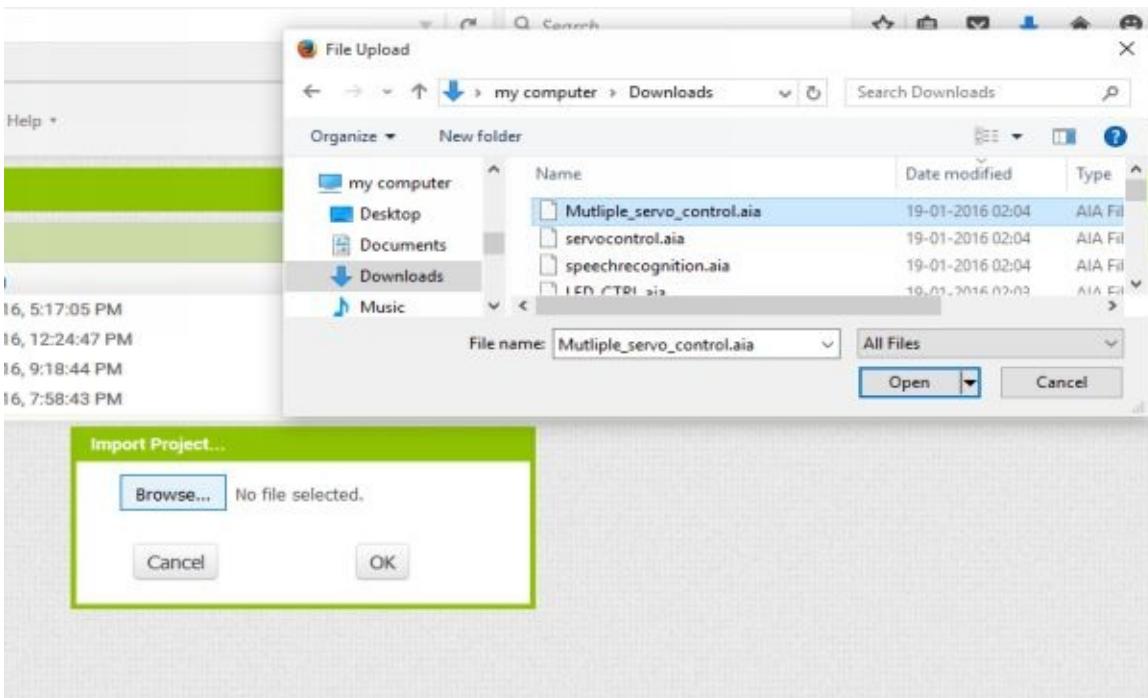
Control the same robot by speech recognition

Use the app created in the beginning chapter and speak to your speech recognition send commands like “forward” “reverse” “left” “right” and “stop” and play with the robot.

How to import aia files to MIT app inventor



Click projects and select import project (.aia) from my computer. Click browse and point to aia file location in your PC or laptop



Contact and app inventor files.

For suggestions and queries you can always mail me at jayakumarmagesh@gmail.com

For all the code and necessary documents for this book please [click here](#)