© 2016 IJSRST | Volume 2 | Issue 6 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology

Bluetooth Communications App Using App Inventor

N. Anju Latha, B. Rama Murthy
Department of Instrumentation, Sri Krishnadevaraya University, Anantapur, Andhra Pradesh, India

ABSTRACT

App Inventor developed by MIT is a visual programming development environment. App Inventor allows the user to create apps for the mobile operating system Android. MIT App Inventor is an innovative programming and app creation that transforms the complex language of text-based coding into visual, drag-and-drop building blocks. In MIT App Inventor, we program using visual programming are called Initial Learning Environments (ILE). It was developed at Google Labs by a team led by MIT's Hal Abelson. Bluetooth is the communications technology. Bluetooth establishes a very low power, short range (up to 10 meters) communications link between two devices. Bluetooth uses the frequency band (2.4 Ghz). Bluetooth use forms of spread spectrum radio links that result in signals moving around within a wide band in ways that enable sharing of the spectrum by multiple devices. In present work, we are creating two separate apps for Bluetooth communications, one is a "server" app that runs on one device, and the other is a "client" app that runs on a second device. In present work the developed app is used for Bluetooth communications and implemented a simple method for sending text data back and forth between two Android devices over the Bluetooth wireless link.

Keywords : MIT App Inventor2, Initial Learning Environments, Bluetooth communication, server, client.

I. INTRODUCTION

Android is a mobile operating system that runs on the Linux Kernel. Android Mobile Application Development is based on Java language codes, as it allows developers to write codes in the Java language. These codes can control mobile devices via Google-enabled Java libraries. It is an important platform to develop mobile applications using the software stack provided in the Google Android SDK. App Inventor is an incredible new system from Google that allows Android applications to be designed and programmed with a Web page and Java interface. The tool of inventor is like a blocks .it split by two parts, one is front screen it give the appearance of the button, image and it looks like an android phone screen. Another part is programming side that is commands. It looks like a c program. The inventor gives real time simulation without android phone. To install the AI starter in your laptop it gives you to simulate real time without phone. It very useful for checking the creating app.

Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz) from fixed and mobile devices and building personal area networks (PANs). Invented by telecom vendor Ericsson in 1994. It was originally conceived as a wireless alternative to RS-232 data cables. It can connect several devices, overcoming problems of synchronization. Bluetooth is managed by the Bluetooth Special Interest Group (SIG), which has more than 25,000 member companies in the areas of telecommunication, computing, networking, and consumer electronics. The IEEE standardized Bluetooth as IEEE 802.15.1.

Bluetooth is a packet-based protocol with a master-slave structure. One master may communicate with up to seven slaves in a piconet. All devices share the master's clock. A master Bluetooth device can communicate with a maximum of seven devices in a piconet (an ad-hoc computer network using Bluetooth technology), though not all devices reach this maximum.

We use classic Bluetooth which is compatible for phones, and is the technology supported by App Inventor. Setting up a Bluetooth devices involves "pairing" the two devices and establishing a connection. There are two separate apps for Bluetooth communications – one is a "server" app that runs on one

device, and the other is a "client" app that runs on a second device. Bluetooth must be enabled on both devices, and the devices need to be paired before running these apps.

II. METHODS AND MATERIAL

Android Application and MIT App Inventor2

Android is an operating system based on the Linux kernel with a user interface based on direct manipulation, designed primarily for touch screen mobile devices such as smart phones and tablet computers. The operating system uses touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard. Despite being primarily designed for touch screen input, it also has been used in televisions, games consoles, digital cameras, and other electronics. We have used MIT App Inventor tool to develop the android application, App Inventor is a cloud-based tool, which means that we can build apps right in our web browser.

In App Inventor, the Designer and Blocks Editor now run completely in the browser we can use App Inventor without downloading anything in computer. we will develop apps on website: ai2.appinventor.mit.edu. To do live testing on our Android device just install the MIT App Inventor Companion app on our Android phone. Then open our project in App Inventor on the web, open the companion on your device, and you can test your apps as you build them Design the App's User Interface by arranging both on and off-screen components. on MIT App Inventor 2 Designer and Blocks Editor.

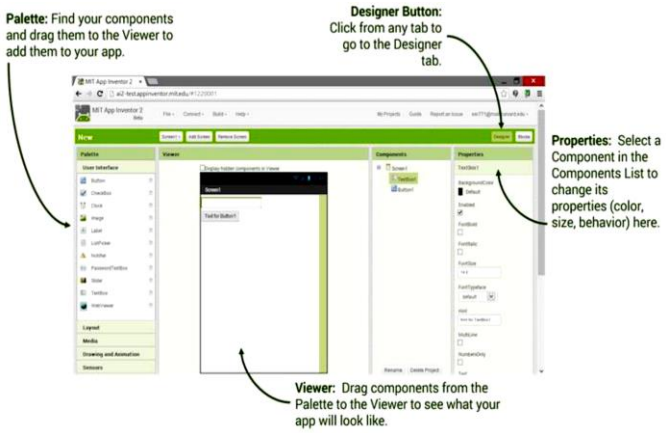

Figure 1: Designer window

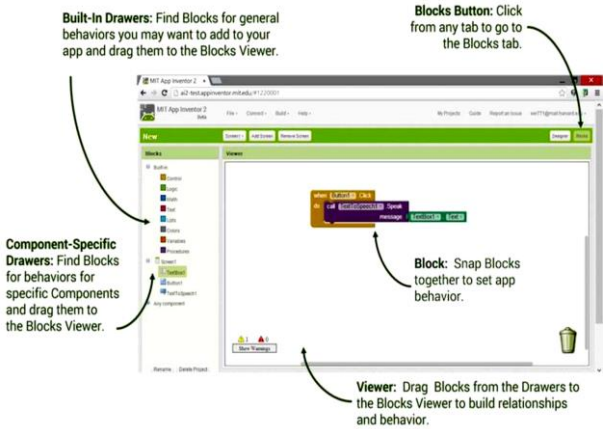

Figure 2: Blocks window

Creating the BlueChat Application

Bluechat Is a Simple chat client/server. With it, two paired devices can send text messages to each other. BlueChat can set up the client/server connection between only two devices. The Bluetooth component in App Inventor is a low-level component. Blue chat having two apps, One is the Bluetooth server, the other the Bluetooth client. Only one of each is needed for two-way connectivity between your devices. However, in the BlueChat application, we use both components so that either device can initiate the connection.

While creating the BlueChat application, we will establish a connection between two Bluetooth devices and then passing a basic text string between them. However, in this project, you strip Bluetooth communication down to its simplest form and use it in its default mode, that is the serial port profile, which emulates a serial connection in sending the data. There are two separate apps for Bluetooth communications – one is a "server" app that runs on one device, and the other is a "client" app that runs on a second device. Bluetooth must be enabled on both devices, and the devices need to be paired before running these apps. The server must be run first on one device and then the client app on the 2nd device connects to the server before data can be sent between the two devices. The Bluetooth server user and client user interface is shown in fig 3. and fig 4.

The main components of the server interface design are:

- Accept Connection Button – press this to set the server to accept a connection from another

device. Connections are not possible until the AcceptConnection service is started.

- Send the following text Button – the text is the following text box is sent to the other Bluetooth device.
- Disconnect Button
- Status messages – Status about the communications link, and any messages received from the other device are shown on the display

Non-visible components – The apps use a clock to cause activities to occur at a preset interval. The Notifier1 component is used to display error messages, and BluetoothServer1 provides the Bluetooth support.

The BluetoothClient1 and BluetoothServer1 components are located in the Connectivity section of the Designer palette

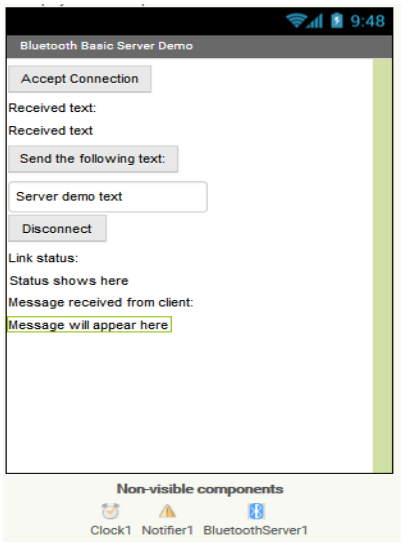

Figure 3. Bluetooth Server design

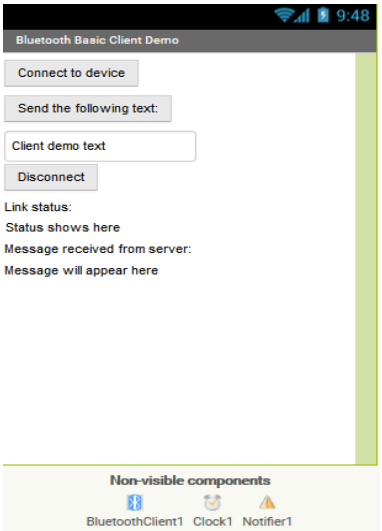

Figure 4. Bluetooth Client design

The user interface is similar to the server except instead of Accept Connection there is a Connect to device button, and instead of a BluetoothServer1 component, the Bluetooth Client1components is used. The Connect to device button is actually a List Picker component and not a standard button. For both the client and server apps, the Timer Interval of the Clock properties is set to 1000 milliseconds or 1 second. Other small values may also be used. This value determines how frequently to check the Bluetooth link for incoming data from the other device. As shown, each app will check the link once per second.

III. RESULT AND DISCUSSION

Blocks Code for Bluetooth Server app

The first step is to check that Bluetooth is activated or switched on. If not, an error message is displayed reminding the user to switch Bluetooth on. The Initialize event occurs when the app is launched and here we check whether Bluetooth is enabled or not. the Bluetooth Server Initialization is shown in fig 5.

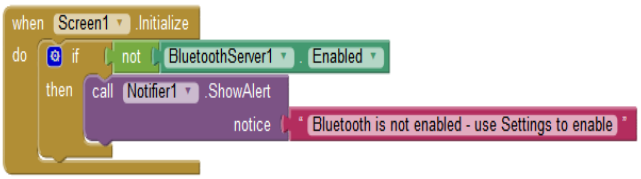

Figure 5. Bluetooth Server Initialization

When the Bluetooth on the device is "on", the next step is to accept a connection from another device, when the btnAcceptConnection button has been pressed. This causes Bluetooth to begin listening for an incoming connection. Once a connection request has been received and processed, a ConnectionAccepted event occurs. Bluetooth Server Accept Connection code is as shown in fig 6.

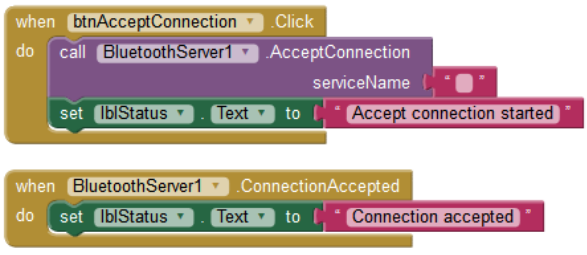

Figure 6. Bluetooth Server Accept Connection

The Timer Event Handles Receiving of Data in Bluetooth Clock1Timer , the clock is set so that the Timer event happens once per second. Every second, the app will check if any data has been received. To prevent reading data when Bluetooth is not connected, an if-then statement checks the IsConnected property of BluetoothServer1. This value is set to true when the devices are connected and false if the connection is not currently available.

The property Bytes Available To Receive tells us how much data is available. If this value is zero, then no data is available. But if the value is greater than zero, then our app may read the incoming data and update the status and messages to the app display. block code for Bluetooth server Clock is as shown in fig 7.

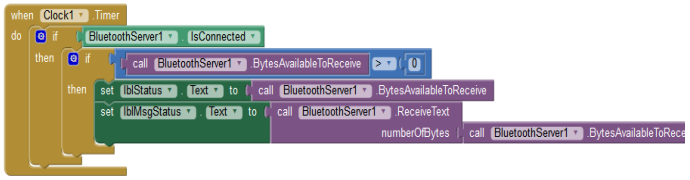

Figure 7. Bluetooth server Clock

The Send Text button is sent data using the SendText method to transmit the data to the other device. The Disconnect button disconnect the connection. Bluetooth server Send Text code and Disconnect code is as shown in fig 8.

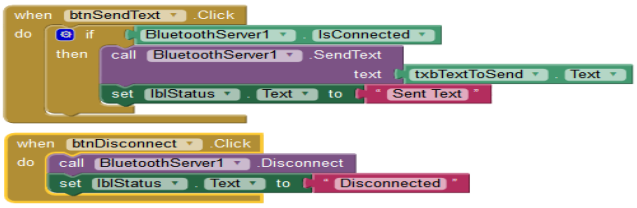

Figure 8. Bluetooth server Send Text code and Disconnect code

Blocks Code for Bluetooth Client App

The client app is same as the server app, but refers to the BlutoothClient1component instead of the BluetoothServer1 component.

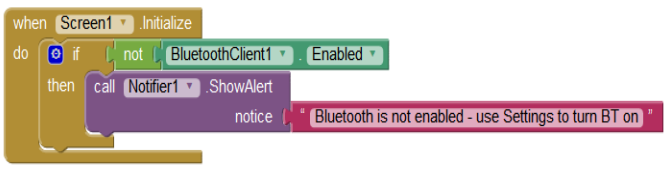

Figure 9. Bluetooth Client Initialization

When the two devices are running, the server app is set up first to accept connections. Then, on the client side, the user selects the Connect ListPicker button and selects the device name from a list of available Bluetooth devices. Because the list of devices is in the form of a list, the ListPicker is a great interface component to display the device list and handle the selection. Bluetooth Client list Picker connect block code is as shown in fig 10.

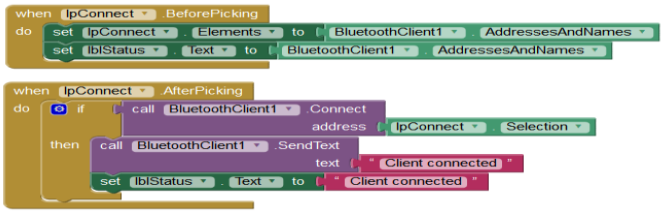

Figure 10. Bluetooth Client list Picker connect

After the device has been selected with the ListPicker user interface, the Connect method of BluetoothClient establishes the connection. The method returns a value of true if the connection was successful in which case a message is sent to the server app.

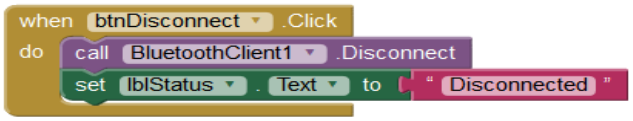

Figure 11. Bluetooth Client Disconnect

Like with the server, the reception of data is implemented using a timer. Once per second, the client checks to see if data is available, and if it is, reads and displays the data on the app display.

While the server must be running prior to the client making a connection, once the two devices are connected, either app can send data to the other app, at any time. The client's error handling is identical to the server's error handling.

App Inventor is a free, open source application that permits people with any level of programming background can create software applications for the Android operating system. App Inventor uses a graphical user interface that allows users to drag and drop blocks. it is very easy to create apps. In this paper we present the basic application code for Bluetooth communications code using App Inventor. Once code is

completed then user can download the source code to android device.

IV. REFERENCES

[1] App Inventor: Create your own Android Apps, published by O'Reilly in 2011.
[2] App Inventor Beginner Tutorials ,MIT center for mobile learning .
[3] Introduction to App Inventor Bluetooth LE "Low Energy" – Part 0, By Edward M • October 3, 2016
[4] http://ai2.appinventor.mit.edu