App Development Laboratory Manual

Introduction to Product Development and Management for Engineers

GNG 2101
Faculty of Engineering
University of Ottawa
Fall 2017

Checks required by TA:

Group Member	User Interface	Programming

Dr. Hanan Anis
Dr. David Knox
Dr. Umar Iqbal

Objective

This lab will introduce students to basic app development software and techniques. Specifically, students will gain hands on design experience by producing a control interface for a user to use to connect to and send control inputs to another device.

Apparatus and Equipment Overview

- 1 x Android device with low energy Bluetooth
- 1 x Personal computer with internet access

Pre-Lab Preparation

Before coming to the lab students should download the MIT AI2 Companion app to an android device from the Google Play store (https://play.google.com/store/apps/details?id=edu.mit.appinventor.aicompanion3). If they do not have an android device, they can download MIT's android emulator instead (they will not be able to connect via Bluetooth). To download the emulator, follow the instructions found here: http://appinventor.mit.edu/explore/ai2/setup-emulator.html. If they do not have an Android device with low energy Bluetooth please ask your TA for a different lab manual with instructions for Iphone or not LE Android devices.

Students must make certain that **BLUETOOTH IS TURNED ON**. Only one phone may be connected to the Bluetooth device at a time.

Enable developer options

1. Tap the Home button to go to your phone's Home screen.
2. Tap the Menu button, then Settings, then Applications.
3. If your phone has an Unknown sources setting, make sure it is checked.
4. Tap Development.
5. Make sure both USB Debugging and Stay Awake are checked.

The steps are similar for newer devices, but the options can be found in different places on the device: On Android 4.0* and newer, these settings are in Settings > Developer options.

Note: On Android 4.2* and newer, Developer options is hidden by default. To make it available, go to Settings > About phone and tap Build number seven times. Return to the previous screen to find Developer options.

*****Unknown Soucres and USB Debugging may be in different directories in Settings depending on the Android version. Use the search button in Settings if you cannot locate them.

Pre-lab Questions

What is the name of the software we will be using?

What is the purpose of this app?

Which communication method does the phone use?

What type of tool do you use to program the app?

What are the two modes available in the software we are using?

Procedure

1. Open MIT App Inventor by going to http://appinventor.mit.edu/explore/ and clicking on the "Create Apps!" button. Select a Gmail account to use (@uottawa.ca accounts work). Once logged in, create and name a new project.
2. In this lab you will to download an extension called **BluetoothLE.axi** from http://appinventor.mit.edu/extensions/ . Now return to the MIT App Inventor and open the extensions tab on the left side of the window and select import extension. Identify and upload the **BluetoothLE.aix** file.
3. Now start creating the user interface. To add an object to the app, drag the corresponding icon from the left hand menu into the prototype build screen. The following objects are required for this introductory app.

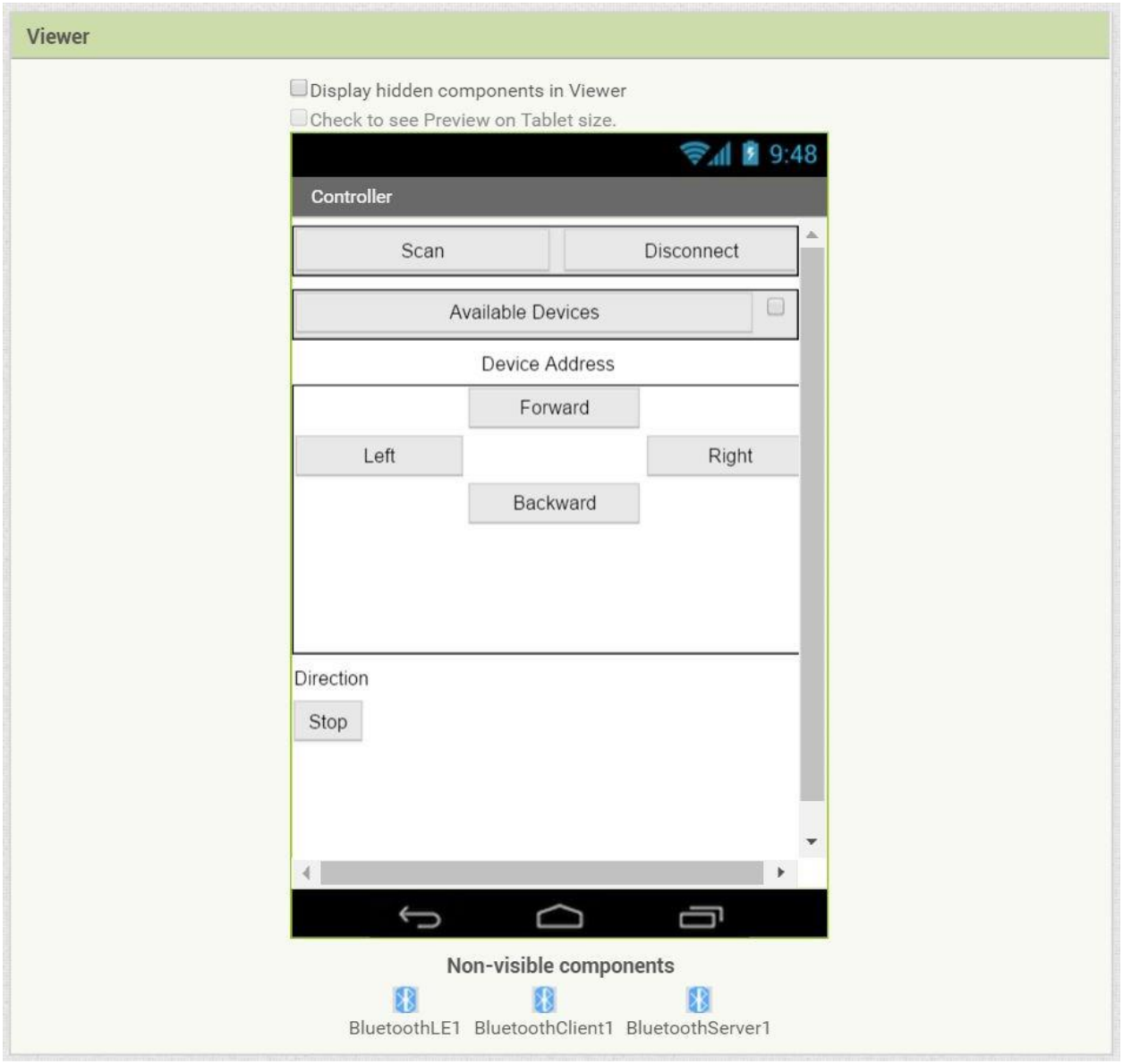
   For Bluetooth connectivity:

   - Two buttons
   - A list picker object
   - A checkbox
   - A label
   - The BluetoothLE extension
   - A regular Bluetooth client and Bluetooth server

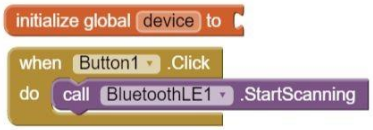
   For control:
   - 4 buttons for directions

- A label
- A stop button

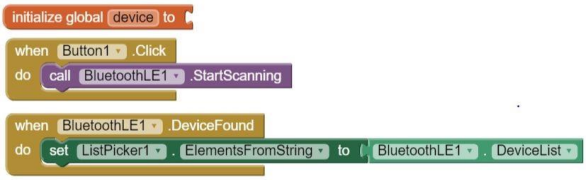
4. Use the layout functions to arrange the input objects. Use the table arrangement for the directional control (hint: use a 3x3 table), and the horizontal arrangement for connectivity buttons.
    - Note: change the size of the tables (fill parent) before adding the buttons
5. Change the default text of the input objects to reflect their intended function. This can be done in the right hand window, once an object (such as a button or label) is selected. Below is an example of a user interface used in this app. Show the TA your completed user interface.



6. Program the app's functionality. Switch to the "Blocks" tab using the button in the top right hand corner. This shows the coding environment used by the app inventor.
7. Add a variable object by selecting "variables" on the left hand side, and initialize a global variable. Change its name to 'device'. This will hold the Bluetooth device's address information once it is connected.
8. Program the first button's function. This button will tell the smartphone to start scanning for Bluetooth devices. On the left hand side, click on 'Button1'. This will open a list of commands a button could be involved with. Select "When Button1.Click", which will create that object in the code window. Select the BluetoothLE1 object in the left window, and scroll down the list. Choose "call BluetoothLE1.StartScanning " and nest this code block inside the Button1.Click block as shown below.

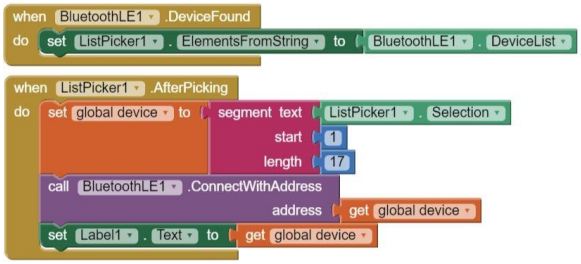


9. Program the list picker element. Choose the "When BluetoothLE1.DeviceFound" option under the BluetoothLE1 object, and nest "set ListPicker1.ElementsFromString to" from the list picker object's menu. Connect the "BluetoothLE1.DeviceList" block (found in the BluetoothLE1 menu) to the end of the list picker block, as seen below. This will display all found devices in a drop down list (the list picker object).



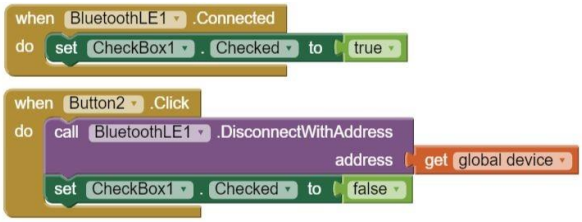
10. Once a device has been selected from the list by the user, it must be connected to by the smartphone. From the list picker command menu, select "when ListPicker1.AfterPicking". This block dictates what happens after a device has been chosen from the list.
11. Create a "set global device to" block from the variables menu on the left. From the "Texts" menu, choose the "segment text" block with the start and length option. Add a "ListPicker1.Selection" block from the ListPicker1 menu to the first attachment point. Create a block with numerical values in it by typing the value and pressing enter. Add a block with a 1 value and a block with a 17 value to the start and length attachment points respectively.
12. Now select a "call BluetoothLE1.ConnectWithAddress" block from the BluetoothLE1 menu, and attach a "get global device" block to the address attachment point. This will connect the smartphone with the Bluetooth device.
13. Select a "set Label1.Text to" block and attach a "get global device" block to it. This will tell the user the address of the Bluetooth device, letting them know that the smartphone is connected to the Bluetooth device. Nest the "set global device to" block, the "call

BluetoothLE1.ConnectWithAddress" block, and the "set Label1.Text to" block into the "when ListPicker1.AfterPicking" block. This will tell the program to run all of these actions immediately after the user selects a Bluetooth device from the list picker. The picture below may be used for reference.

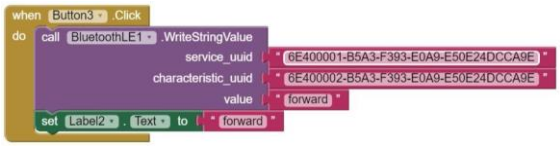


14. Program the checkbox to activate when a device has been connected to the phone. Select the "when BluetoothLE1.Connected" block from the BluetoothLE1 menu and nest a "set Checkbox1.Checked to" block into it (found under the Checkbox1 menu). Type "true" and press enter, and connect the created "true" block to the Checkbox1.Checked block, as seen below.

15. Program Button2 to act as a disconnect button. Bring a "When Button2.Click" block into the code area, and nest a "set CheckBox1.checked to" with a "false" block attached into it. Select a "call BluetoothLE1.DisconnectWithAddress" block and nest it above the CheckBox1.Checked block within the Button1.Click block as seen below. Add a "get global device" block to the BluetoothLE1.Disconnect block. This will disconnect the Bluetooth device from the smartphone, and uncheck the checkbox.



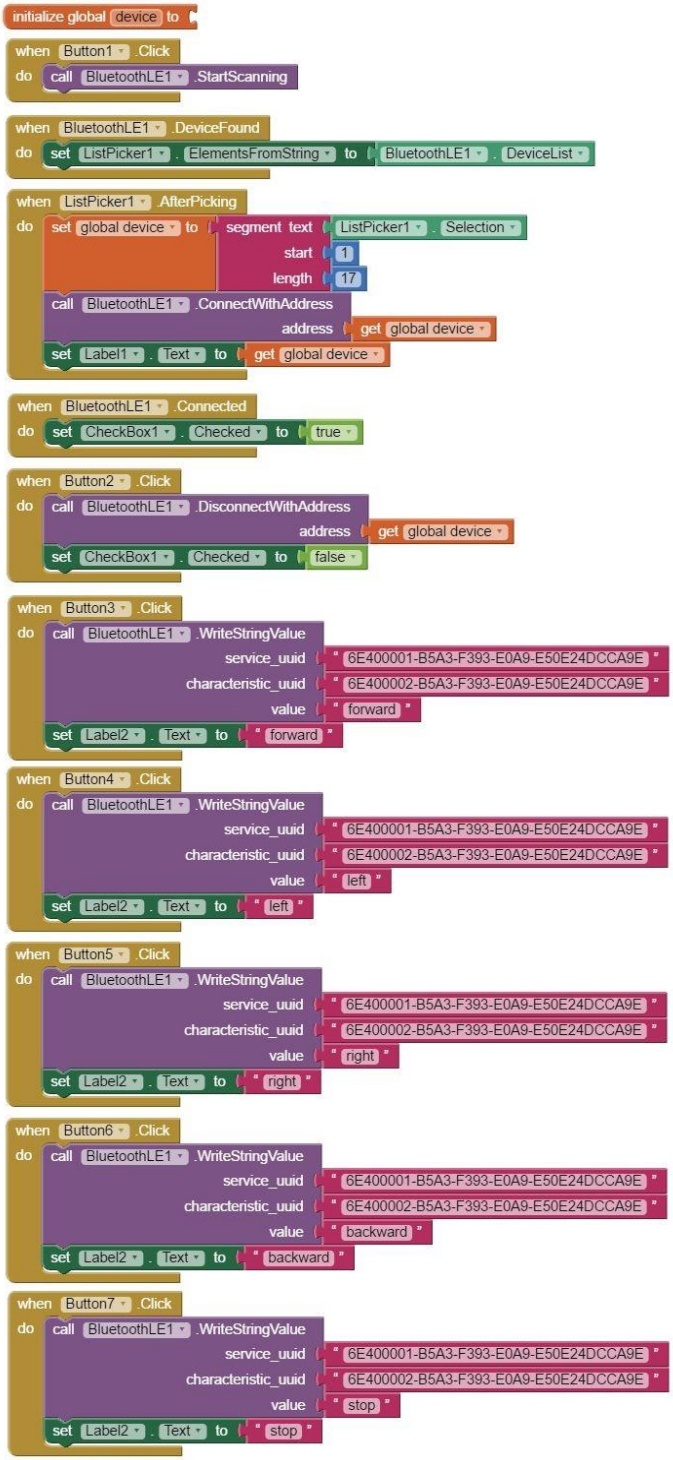
16. Program the directional control buttons, label, and stop button. One button should be programed for each direction to send a string variable to the Bluetooth device (either "forward", "backward", "left", or "right"). The label should display to the user the last button that was pressed, and the final button should be programmed to send a string variable containing "stop" to the Bluetooth device. Below is an example of the code to program the forward button. It can be duplicated and altered for the other directions and the stop button. Be sure to use the same service UUID and characteristic UUID, as these must match the UUID's on the Bluetooth device for communication to be established.



Service UUID: 6E400001-B5A3-F393-E0A9-E50E24DCCA9E

Characteristic UUID: 6E400002-B5A3-F393-E0A9-E50E24DCCA9E

This is all the programming required for this app to run. Note that there are locations that could be optimized, especially when disconnecting and reconnecting to Bluetooth devices. When using this app in the lab, close and restart the app if anything doesn't seem to work and try again before asking the T.A. for help. The code below is a complete example of the app programming section, and may be used for troubleshooting.



17. To test the app on an android device, download and save the app as an .apk file (select "Build" at the top of the webpage). Email the file to an email address available on the phone and open the file. You may have to allow the phone to install third-party applications. The app should then install, and can be run as a normal app when finished. You may also prompt MIT App Inventor to display a QR code. Scan the code with the

android phone (MIT AI2 Companion), and the app should download automatically. Show the TA your work.

Additional resources

- To learn more about MIT app inventor you can follow some beginner tutorials at http://appinventor.mit.edu/explore/ai2/beginner-videos.html.
- For more complicated app development you can try Android studio (which is installed on the university computers), here is a guide https://developer.android.com/training/index.html.