COMP4336/9336

Device to device communication 2 (Bluetooth)

Lab Objectives

- Learn how to check the status of Bluetooth whether it is enabled or disabled.
- Learn how to enable/disable the Bluetooth interface.
- Learn how to discover available devices with *on* Bluetooth.
- Learn how to connect to a nearby device such as a mobile phone or laptop via Bluetooth.
- Learn how to exchange message between two devices via Bluetooth.

Preparation

- *Background*: Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength radio transmissions in the ISM band from 2400–2480 MHz) from fixed and mobile devices. You can find numerous short introductions and tutorials of different enhancement of Bluetooth from the Internet (e.g. http://en.wikipedia.org/wiki/Bluetooth) or the computer networking books published in recent years.

- *Android facilities to manage Bluetooth*: In Android development, you can programmatically configure the Bluetooth interface of you android devices. Android comes with a number of system services that are available for Android developers. Android Development, like all other systems services, provides an API for managing all aspects of the Bluetooth via *android.bluetooth* package. The Bluetooth API supports both "Classic Bluetooth" and Bluetooth Low Energy.

- *Useful Android classes* to do the lab tasks:

  - **BluetoothAdapter**

    Represents the local Bluetooth adapter. The BluetoothAdapter is the entry-point for all Bluetooth interaction. Using this, you can discover other Bluetooth devices, query a list of bonded (paired) devices, instantiate a Bluetooth Device using a known MAC address, and create a Bluetooth ServerSocket to listen for communications from other devices. More details:

    http://developer.android.com/reference/android/bluetooth/BluetoothAdapter.html

- **BluetoothDevice**

  Represents a remote Bluetooth device. Use this to request a connection with a remote device through a Bluetooth Socket or query information about the device such as its name, address, class, and bonding state. More details:

  http://developer.android.com/reference/android/bluetooth/BluetoothDevice.html

- **BluetoothSocket**

  Represents the interface for a Bluetooth socket (similar to a TCP Socket). This is the connection point that allows an application to **exchange data** with another Bluetooth device via InputStream and OutputStream. More details:

  http://developer.android.com/reference/android/bluetooth/BluetoothSocket.html

- **BroadcastReceiver**

  Base class for code that will receive intents sent by sendBroadcast(). You will need this class when you want to get the list of discovered nearby Bluetooth devices. More details:

  http://developer.android.com/reference/android/content/BroadcastReceiver.html

- *Some useful notes* which might be helpful to accomplish tasks:

  - *Take required permissions*: You need to add some privileges to the *Manifest File* in order to allow the application access to the Bluetooth:

    ```
    <uses-permission
    android:name="android.permission.BLUETOOTH"/>
    <uses-permission
    android:name="android.permission.BLUETOOTH_ADMIN"/>
    ```

    Note: The element types in the manifest are ordered. Uses-permission needs to be first under the tag or outside of the application element otherwise the phone does not allow to access the Bluetooth interface and the program will crash.

  - *Checking hardware:*

    With this line of code you can check whether your device supports Bluetooth or not:

    ```
    BluetoothAdapter btAdapter = BluetoothAdapter.getDefaultAdapter();
    if (btAdapter == null) { /* Inform user that device does not
    support Bluetooth*/}
    ```

- *Enabling Bluetooth Interface*: The Bluetooth interface should be enabled before testing your program. To enable Bluetooth manually, go to *Setting* -> Bluetooth->*Enabled.* You can do that via application such as:

```
if (!btAdapter.isEnabled()) {  btAdapter.enable(); }
```

- *Discover nearby devices:*

  You can use *startDiscovery* methods from *BluetoothAdapter* class to discover the available nearby devices. The process is asynchronous and the method will immediately return with a *boolean* indicating whether discovery has successfully started. The discovery process usually involves an inquiry scan of about 12 seconds, followed by a page scan of each found device to retrieve its Bluetooth name.

  Your application must register a *BroadcastReceiver* for the ACTION_FOUND Intent in order to receive information about each device discovered. For each device, the system will broadcast the ACTION_FOUND Intent. This Intent carries the extra fields EXTRA_DEVICE and EXTRA_CLASS, containing a *BluetoothDevice* and a BluetoothClass, respectively. For example, here's how you can register to handle the broadcast when devices are discovered:

```java
// Create a BroadcastReceiver for ACTION_FOUND
private final BroadcastReceiver mReceiver = new
BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        // When discovery finds a device
        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            // Get the BluetoothDevice object from the Intent
            BluetoothDevice device =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            // Add the name and address to an array adapter to
show in a ListView
     mArrayAdapter.add(device.getName() + "\n"
+device.getAddress());
        }
    }
};
// Register the BroadcastReceiver
IntentFilter filter = new
IntentFilter(BluetoothDevice.ACTION_FOUND);
registerReceiver(mReceiver, filter);
```

- *Connecting Devices*: In order to create a connection between two mobile phones actually you have to establish connection between your applications on two devices. Hence, you must implement both the server-side and client-side mechanisms, because one device must open a server socket and the other one must initiate the connection.

  Note: You have to make both phone paired before start to run your program. You can do that via Bluetooth setting of your phone.
  Details and example to connect as a server/client:
  http://developer.android.com/guide/topics/connectivity/bluetooth.html#ConnectingDevices

- *Sample code for optional task:* there is comprehensive code to construct a simple peer-to-peer messaging system that works between two paired Bluetooth devices. It would be useful if you read and try to understand that before your lab session.
  http://manojprasaddevelopers.blogspot.com.au/2012/02/bluetooth-data-transfer-example.html

Lab Tasks

Task 1 (0.5 Marks): Check the status of Bluetooth
Develop a program to check the availability of Bluetooth on your mobile phone following by checking the status whether it is enabled or not. The status of the Bluetooth interface should be displayed in a *text view*.

Task 2 (0.5 Marks): Bluetooth enabled device discovery
Develop a program to search and discover all the nearby devices which their Bluetooth interfaces is *enabled* and show the device name and MAC address in a *list view*. The program should repeat the discovery process when user clicks the "*discover*" button.
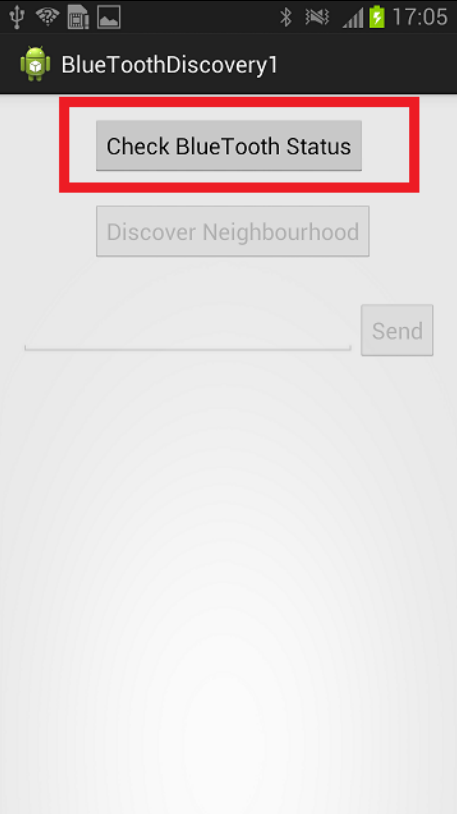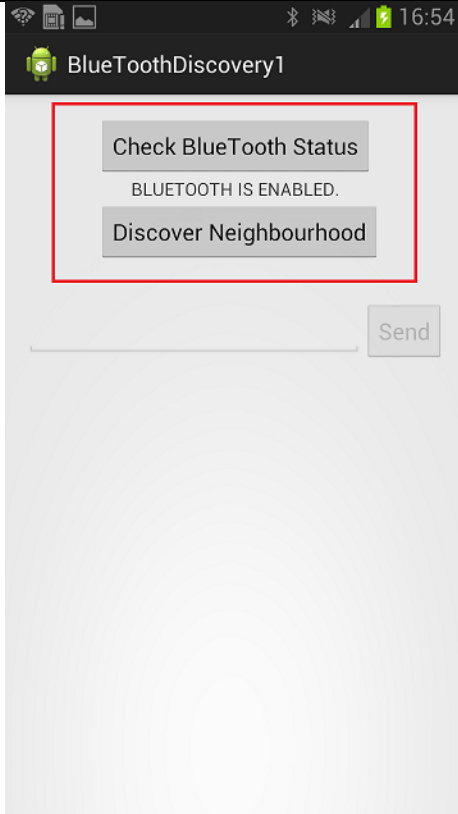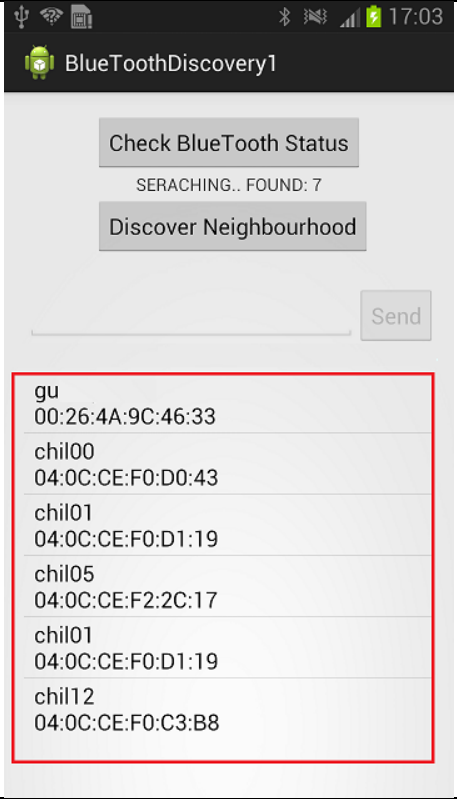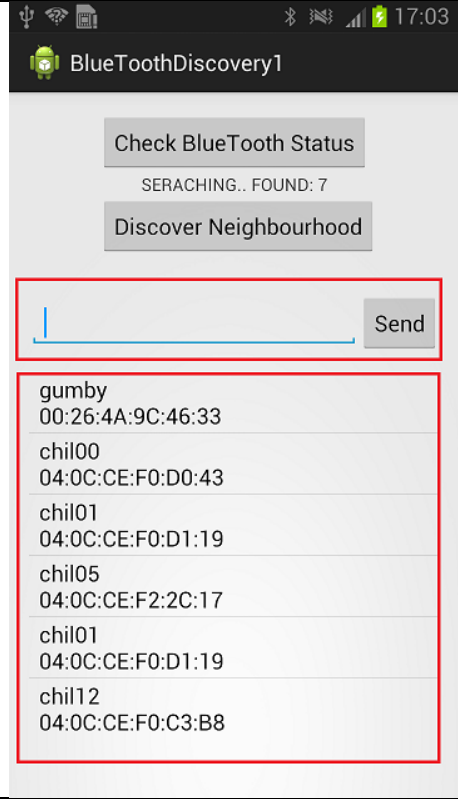
Task 3 (0.5 Marks): Connect to a nearby device.
Based on the program developed in task2, add a new functionality to your program in which user can select one of the discovered device and connect to that after the target device confirms your request. For that you can ask one of your classmates to cooperate with you.

Task4 (0.5 Marks): A simple text messenger via Bluetooth

Develop a program to connect to one of your nearby mobile phones via Bluetooth and then exchange message via Bluetooth. Note that your program should be installed on both mobile phones.

Task1 (1)	Task1 (2)
Task2	Task3/4