Bluetooth API Implementation into Android

Sergey Konev[1], *Anastasia* Stus[1], *Elena* Kasyanenko[1] and *Vasiliy* Dolgov[1]

[1]Don State Technical University, Chair "Production Process Automation", 344000, Gagarin square, Rostov-on-Don, Russia

Abstract. Bluetooth is a popular method of communication between devices. Many smartphones today have the capability to communicate using Bluetooth. Android developers sometimes need to use Bluetooth in their projects. Android OS provides a powerful API for Bluetooth that allows to simplify scanning the environment for devices, pairing and connecting, data transfer between devices and more. However, utilizing the Bluetooth API can be difficult for first-time users. The objective of this article is to demonstrate the key points of implementing Bluetooth API in the Android application.

1 Introduction

Bluetooth API is located in the android.bluetooth package [1]. It contains more than ten classes. The package includes specific classes for medical Bluetooth devices, audio headsets and data transfer. This article will only deal with the basic classes implementing Bluetooth basic functionality; without them, it is impossible to create any Bluetooth-supporting application [2]:

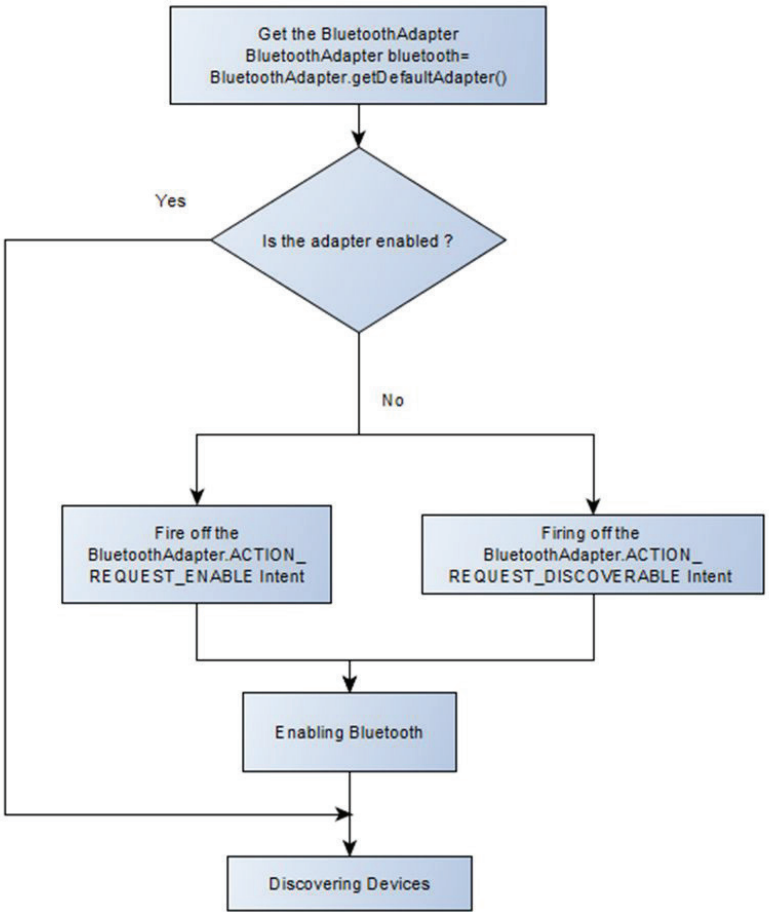

Fig. 1. Enabling Bluetooth flowchart

* Corresponding author: lex1998@rambler.ru

- The Bluetooth Adapter class represents the Bluetooth radio hardware on the

local device. The Bluetooth Adapter is the entry-point for all Bluetooth interaction.

- The Bluetooth Device class represents a remote Bluetooth device.

- The Bluetooth Server Socket class is used to open a socket to listen for incoming connections and provides a Bluetooth Socket object when a connection is made. Represents an open server socket that listens for incoming requests (similar to a TCP Server Socket).

- The Bluetooth Socket class is used by the client to establish a connection to a remote device.

Let's start with the description of the process of the application development for Android. We will study the main stages of Bluetooth implementation in Android application in more details

2 Enabling Bluetooth

At the very beginning of the work, before the connection, it is necessary to check whether your device is equipped with a Bluetooth unit. If the answer is positive, the user may be offered to enable the unit or make the device visible for other users, and that will enable it automatically. The test of the unit and its performance, as well as enabling is carried out with the use of methods of Bluetooth Adapter class (Fig. 1).

3 Discovering devices

The search of devices for connection is also performed with the use of Bluetooth Adapter class. You can also start scanning or request the list of paired, i.e. already known devices.

The search of new available Bluetooth devices is performed during scanning. If a device with Bluetooth enabled is within the reach, it will send some identifying information in response to the request: name, class, MAC-address. Based on this information you can arrange connection and data transfer. To start scanning, you need to call the start Discovery method. Scanning is performed in a separate asynchronous flow. This method requires the BLUETOOTH_ADMIN permission. To retrieve information on the devices found, the application has to register a broadcasting receiver for the following Intents:

ACTION_DISCOVERY_STARTED: Occurs when the discovery process initiates;

ACTION_FOUND: Occurs each time a remote Bluetooth device is found;

ACTION_DISCOVERY_FINISHED: Occurs when the discovery process completes.

The search requires many resources, so as soon the suitable device has been found, it is necessary to stop the scanning process. Besides, the search process may also narrow gating width of the radio channel significantly, that's why it's better to refrain from search of new devices when the connection is established.

Before you start the environment scanning, it's reasonable to show the list of already known and paired devices to the user. It is necessary to make difference between paired and connected devices. Paired devices have a reference key, which they can use for authentication, and they are able to create an encrypted connection with each other. Connected devices share one radio channel and are able to transfer data to each other. You can use the Bluetooth Adapter to query for available Bluetooth devices to connect to. The get Bonded Devices method returns a set of Bluetooth Device objects that represent the list of paired devices.

4 Establishing connections between devices

Connection through Bluetooth is arranged according to the "client-server" principle and is similar to the connection through TCP-sockets [3]. To connect two devices, server and client parts of the code have to be written. One of the devices shall open a listening server socket for waiting the request for incoming connection, and the second shall initiate the connection, i.e. to give the client-socket back using MAC-address of the server. Server and client are considered connected when both of them have an active Bluetooth Socket object on one and the same radio channel [4]. After this data exchange can be initiated.

To set up a server socket and accept a connection, complete the following sequence of steps:

1. Get a Bluetooth Server Socket by calling listen Using Rfcomm With Service Record. The string is an identifiable name of your service, which the system automatically writes to a new Service Discovery Protocol (SDP) database entry on the device. The name is arbitrary and can simply be your application name. The UUID is also included in the SDP entry and forms the basis for the connection agreement with the client device.
2. Start listening for connection requests by calling method accept. This is a blocking call. It returns when either a connection has been accepted or an exception has occurred. A connection is accepted only when a remote device has sent a connection request containing a UUID that matches the one registered with this listening server socket. When successful, accept returns a connected Bluetooth Socket.
3. Unless you want to accept additional connections, call method close. This method call releases the server socket and all its resources, but doesn't close the connected Bluetooth Socket that's been returned by accept.

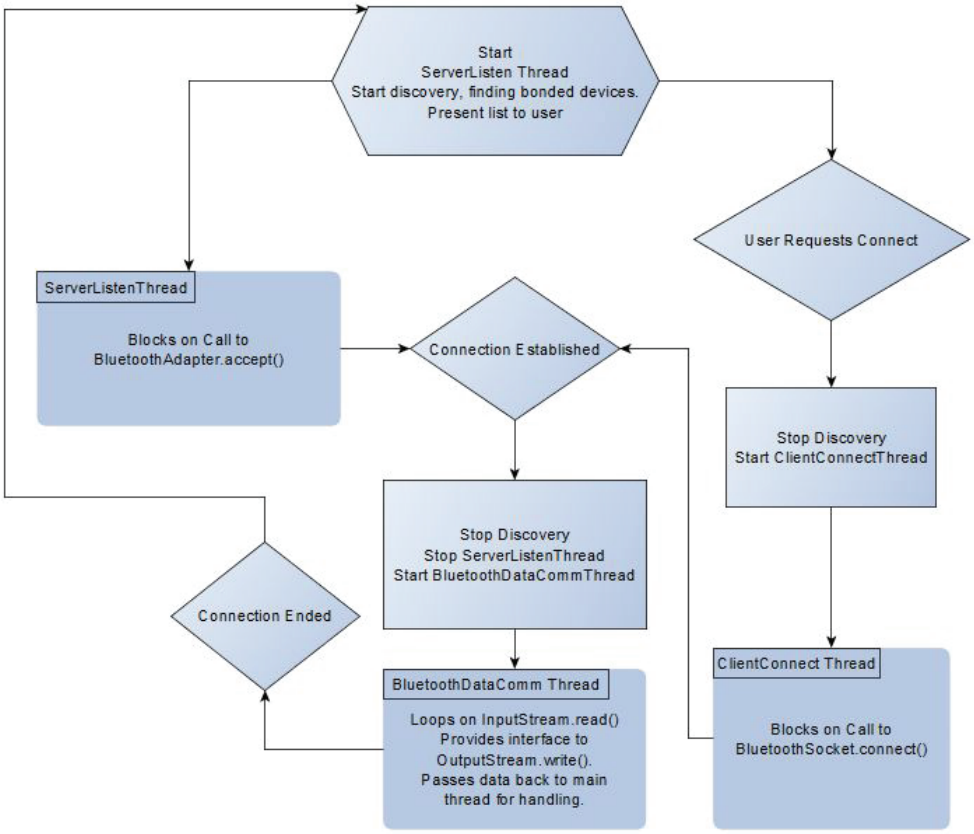

Fig. 2. Diagram showing behaviour flow for a Bluetooth application on Android

In order to initiate a connection with a remote device that is accepting connections on an open server socket, client needs to obtain a Bluetooth Device object that represents the remote device. Consider now the steps of connecting to the server from the client's point of view:

1. Using the Bluetooth Device, get a Bluetooth Socket by calling create Rfcomm Socket To Service Record (UUID). The UUID passed here must match the UUID used by the server device when it called listen Using Rfcomm With Service Record (String, UUID) to open its Bluetooth Server Socket.
2. Initiate the connection by calling connect. Note that this method is a blocking call. After a client calls this method, the system performs an SDP lookup to find the remote device with the matching UUID. If the lookup is successful and the remote device accepts the connection, it shares the RFCOMM channel to use during the connection, and the connect method returns. The method connect is a blocking call, you should always perform this connection procedure in a thread that is separate from the main activity (UI) thread.

After a successful connection, each of the connected devices has a Bluetooth Socket object with which it is easy to implement reception and transmission of data:

1. Using the get Input Stream and get Output Stream methods, get the Input Stream and Output Stream objects that control the transmission through the socket.
2. Read and write data to the streams using read (byte) and write (byte).

Fig. 2 shows a reasonable layout for a Bluetooth implementation

4 Conclusion

Bluetooth provides users with a quick and easy way of exchanging data between a wide range of different devices. Bluetooth technology is a robust, easy-to-use wireless solution for mobile applications. Utilizing Bluetooth in Android applications can be daunting for those unfamiliar with the process. This article can help Android developers gain experience in developing an Android application with Bluetooth.

References

1. Android's Bluetooth API developer guide. Retrieved September 19, (2017).
2. Conder S., Darcey L. *Android Wireless Application Development. Second edition*. (Addison-Wesley, 2011).
3. Gratton D.A. *The Handbook of Personal Area Networking Technologies and Protocols*. (Cambridge University Press, Cambridge, 2013).
4. Bluetooth Core Specification v.5.0. Retrieved September 19 (2017).