Bluetooth Chat HelloWorld

Bluetooth HelloWorld (Chat example)

This Java ME Bluetooth chat example enables server-client communication over L2CAP protocol. It is separated in 3 classes: a main midlet to start application, a Bluetooth Server class and a Bluetooth Client class.

The midlet: BtoothChat.java

```
/*
 * BtoothMidlet.java
 *
 * Created on April 18, 2007, 12:39 AM
 */

package wiki.nokia.example;

import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

/**
 *
 * @author  Luis Albinati (luis.albinati@gmail.com)
 */
public class BtoothChat extends MIDlet implements CommandListener {

    private Display display;
    private List list;


    public BtoothChat() {

        display = Display.getDisplay(this);
        list = new List("Select", List.EXCLUSIVE);
        list.append("Server", null);
        list.append("Client", null);

        Command cmd_ok = new Command("OK", Command.OK, 1);

        list.addCommand(cmd_ok);
        list.setCommandListener(this);
    }

    public void startApp() {
        display.setCurrent(list);
    }

    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {
    }

    public void commandAction(Command command, Displayable d) {

        if (command.getCommandType()==Command.OK) {
            switch (list.getSelectedIndex()){
                case 0: {

                    new BluetoothServer(this); //starting instance of a server

                    break;
                }

                case 1: {

                    new BluetoothClient(this); //starting instance of a client

                    break;
                }
            }
        }

    }
    protected void setAlert(String info) {
        Alert a = new Alert("INFO");
        a.setString(info);
        a.setTimeout(Alert.FOREVER);
        display.setCurrent(a);
    }
}
```

The server class: BluetoothServer.java

```
/*
 * BluetoothServer.java
 *
 * Created on April 18, 2007, 12:45 AM
 *
 */

package wiki.nokia.example;

import java.io.IOException;
import javax.bluetooth.BluetoothStateException;
```


(C) Copyright Nokia 2013. All rights reserved.

```
import javax.bluetooth.DiscoveryAgent;
import javax.bluetooth.L2CAPConnection;
import javax.bluetooth.L2CAPConnectionNotifier;
import javax.bluetooth.LocalDevice;
import javax.microedition.io.Connector;

/**
 *
 * @author Luis Albinati (luis.albinati@gmail.com)
 */
public class BluetoothServer implements Runnable {

    private boolean listening = true;
    private LocalDevice local_device;
    private BtoothChat midlet;
    private String deviceName;
    private L2CAPConnection con;

    /** Creates a new instance of BluetoothServer */
    public BluetoothServer(BtoothChat midlet) {
        this.midlet = midlet;
        Thread t = new Thread(this);
        t.start();
    }

    public void run(){
        System.out.println("Starting server - please wait...");

        try {
            local_device = LocalDevice.getLocalDevice();
            DiscoveryAgent disc_agent = local_device.getDiscoveryAgent();
            local_device.setDiscoverable(DiscoveryAgent.LIAC);
            String service_UUID = "00000000000010008000006057028A06";
            deviceName = local_device.getFriendlyName();
            String url = "btl2cap://localhost:" + service_UUID + ";name=" + deviceName;

            L2CAPConnectionNotifier notifier = (L2CAPConnectionNotifier)Connector.open(url);
            con = notifier.acceptAndOpen();

            while (listening) {
                if (con.ready()){
                    byte[] b = new byte[1000];
                    con.receive(b);
                    String s = new String(b, 0, b.length);
                    System.out.println("Recieved from client: " + s.trim());
                    midlet.setAlert(s.trim());
                    send("Hello client, my name is: " + getName());
                    listening=false;
                }
            }

        } catch(BluetoothStateException e){System.out.println(e);} catch(IOException f){System.out.println(f);}
    }
    private void send(String s){
        byte[] b = s.getBytes();
        try {
            con.send(b);
        } catch(IOException e){
            System.out.println(e);
        }
    }
    private String getName(){
        return deviceName;
    }
}
```

The client class: BluetoothClient.java

```
/*
 * BluetoothClient.java
 *
 * Created on April 18, 2007, 12:45 AM
 *
 */

package wiki.nokia.example;

import java.io.IOException;
import java.util.Enumeration;
import java.util.Vector;
import javax.bluetooth.BluetoothStateException;
import javax.bluetooth.DeviceClass;
import javax.bluetooth.DiscoveryAgent;
import javax.bluetooth.DiscoveryListener;
import javax.bluetooth.L2CAPConnection;
import javax.bluetooth.LocalDevice;
import javax.bluetooth.RemoteDevice;
import javax.bluetooth.ServiceRecord;
import javax.bluetooth.UUID;
import javax.microedition.io.Connector;

/**
 *
 * @author  Luis Albinati (luis.albinati@gmail.com)
 */
public class BluetoothClient implements Runnable {

    private InquiryListener inq_listener;
    private ServiceListener serv_listener;
    private boolean listening = true;
    private BtoothChat midlet;
    private String deviceName;
    private L2CAPConnection con;

    /** Creates a new instance of BluetoothClient */
    public BluetoothClient(BtoothChat midlet){
        this.midlet = midlet;
        Thread t = new Thread(this);
        t.start();
    }
    public void run() {
        System.out.println("Starting client - please wait...");
        try {
            LocalDevice local_device = LocalDevice.getLocalDevice();
```


(C) Copyright Nokia 2013. All rights reserved.

```
            DiscoveryAgent disc_agent = local_device.getDiscoveryAgent();
            local_device.setDiscoverable(DiscoveryAgent.LIAC);
            inq_listener = new InquiryListener();
            synchronized(inq_listener) {
                disc_agent.startInquiry(DiscoveryAgent.LIAC, inq_listener);
                try {inq_listener.wait(); }  catch(InterruptedException e){}
            }

            Enumeration devices = inq_listener.cached_devices.elements();

            UUID[] u = new UUID[]{new UUID( "00000000000010008000006057028A06", false )};
            int attrbs[] = { 0x0100 };
            serv_listener = new ServiceListener();
            while( devices.hasMoreElements() ) {
                synchronized(serv_listener) {
                    disc_agent.searchServices(attrbs, u, (RemoteDevice)devices.nextElement(), serv_listener);
                    try {serv_listener.wait();} catch(InterruptedException e){}
                }
            }
        } catch (BluetoothStateException e) {System.out.println(e);}

        if (serv_listener.service!=null){
            try {
                String url;
                url = serv_listener.service.getConnectionURL(0, false);
                deviceName = LocalDevice.getLocalDevice().getFriendlyName();
                con = (L2CAPConnection) Connector.open( url );
                send("Hello server, my name is: " + getName());

                byte[] b = new byte[1000];
                while (listening) {
                    if (con.ready()){
                        con.receive(b);
                        String s = new String(b, 0, b.length);
                        System.out.println("Received from server: " + s.trim());
                        midlet.setAlert(s.trim());
                        listening = false;
                    }
                }
            } catch (IOException g) {System.out.println(g);}
        }
    }
    private void send(String s){
        byte[] b = s.getBytes();
        try {
            con.send(b);
        } catch(IOException e){
            System.out.println(e);
        }
    }
    private String getName(){
        return deviceName;
    }
}
class InquiryListener implements DiscoveryListener {
    public Vector cached_devices;
    public InquiryListener() {
        cached_devices = new Vector();
    }

    public void deviceDiscovered( RemoteDevice btDevice, DeviceClass cod ) {
        int major = cod.getMajorDeviceClass();
        if( ! cached_devices.contains( btDevice ) ) {
            cached_devices.addElement( btDevice );
        }
    }

    public void inquiryCompleted( int discType ) {
        synchronized(this){ this.notify(); }
    }

    public void servicesDiscovered( int transID, ServiceRecord[] servRecord ) {}
    public void serviceSearchCompleted( int transID, int respCode ) {}
}

class ServiceListener implements DiscoveryListener {
    public ServiceRecord service;
    public ServiceListener() { }

    public void servicesDiscovered( int transID, ServiceRecord[] servRecord ) {
        service = servRecord[0];
        System.out.println("foundService");
    }

    public void serviceSearchCompleted( int transID, int respCode ) {
        synchronized( this ){ this.notify();}
    }

    public void deviceDiscovered( RemoteDevice btDevice, DeviceClass cod ){}
    public void inquiryCompleted( int discType ){}
}
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.