AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="aula07_bate_papo_bluetooth.android.pdm.aula07_batepapo_bluetooth" >

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".BluetoothChatActivity"
            android:label="@string/app_name"
            android:configChanges="orientation|keyboardHidden|keyboard|screenSize">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

res/layout/activity_bluetooth_chat.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <ListView
        android:id="@+id/lstHistorico"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_weight="1" />
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <EditText
            android:id="@+id/edtMsg"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1" />
        <Button
            android:id="@+id/btnEnviar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Enviar" />
    </LinearLayout>
</LinearLayout>
```

res/menu/menu_bluetooth_chat.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item android:id="@+id/action_servidor"
        android:title="@string/acao_iniciar_servidor"
        app:showAsAction="never" />
    <item android:id="@+id/action_cliente"
        android:title="@string/acao_iniciar_cliente"
        app:showAsAction="never" />
</menu>
```

res/values/strings.xml

```
<resources>
    <string name="app_name">"Aula07-Batepapo-Bluetooth "</string>

    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>

    <string name="acao_iniciar_servidor">Iniciar servidor</string>
    <string name="acao_iniciar_cliente">Iniciar cliente</string>
    <string name="msg_erro_bt_indisponivel">Aparelho não suporta Bluetooth</string>
    <string name="msg_procurando_dispositivos">Procurando dispositivos...</string>
    <string name="msg_ativar_bluetooth">Você deve ativar o Bluetooth para continuar.</string>
    <string name="msg_aparelho_invisivel">Para iniciar o servidor, seu aparelho deve estar
visível.</string>
    <string name="aparelhos_encontrados">Aparelhos encontrados</string>
    <string name="aguarde">Aguarde</string>
    <string name="mensagem_servidor">Aguardando por conexões...</string>
    <string name="msg_desconectou">Desconectou</string>
</resources>
```

BluetoothChatActivity.java

```
package aula07_bate_papo_bluetooth.android.pdm.aula07_batepapo_bluetooth;

import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothServerSocket;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.support.v7.app.ActionBarActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

public class BluetoothChatActivity extends ActionBarActivity
        implements View.OnClickListener, DialogInterface.OnClickListener,
        DialogInterface.OnCancelListener {

    private static final String SERVICO = "DominandoChat";
    private static final UUID MEU_UUID =
            UUID.fromString("2accaffd-18dd-43ac-a2c4-623550cf9c8f");

    private static final int BT_TEMPO_DESCOBERTA = 30;
    private static final int BT_ATIVAR = 0;
    private static final int BT_VISIVEL = 1;

    private static final int MSG_TEXTO = 0;
    private static final int MSG_DESCONECTOU = 2;

    private ThreadServidor mThreadServidor;
    private ThreadCliente mThreadCliente;
    private ThreadComunicacao mThreadComunicacao;

    private BluetoothAdapter mAdaptadorBluetooth;
    private List<BluetoothDevice> mDispositivosEncontrados;
    private EventosBluetoothReceiver mEventosBluetoothReceiver;

    private DataInputStream is;
    private DataOutputStream os;

    private ArrayAdapter<String> mMensagens;
    private TelaHandler mTelaHandler;
    private ProgressDialog mAguardeDialog;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bluetooth_chat);

        mTelaHandler = new TelaHandler();
        mMensagens = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1);

        ((ListView) findViewById(R.id.lstHistorico)).setAdapter(mMensagens);

        mEventosBluetoothReceiver = new EventosBluetoothReceiver();
        mDispositivosEncontrados = new ArrayList<BluetoothDevice>();
        mAdaptadorBluetooth = BluetoothAdapter.getDefaultAdapter();

        if (mAdaptadorBluetooth != null) {
            if (!mAdaptadorBluetooth.isEnabled()) {
                Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
```

```
                startActivityForResult(enableBtIntent, BT_ATIVAR);
            }

        } else {
            Toast.makeText(this, R.string.msg_erro_bt_indisponivel, Toast.LENGTH_LONG).show();
            finish();
        }

        IntentFilter filter1 = new IntentFilter(BluetoothDevice.ACTION_FOUND);
        IntentFilter filter2 = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);

        registerReceiver(mEventosBluetoothReceiver, filter1);
        registerReceiver(mEventosBluetoothReceiver, filter2);

        findViewById(R.id.btnEnviar).setOnClickListener(this);
    }

    protected void onDestroy() {
        super.onDestroy();
        unregisterReceiver(mEventosBluetoothReceiver);
        paraTudo();
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_bluetooth_chat, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_cliente:
                mDispositivosEncontrados.clear();
                mAdaptadorBluetooth.startDiscovery();
                exibirProgressDialog(R.string.msg_procurando_dispositivos, 0);
                break;

            case R.id.action_servidor:
                Intent discoverableIntent = new Intent(
                        BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
                discoverableIntent.putExtra(
                        BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION,
                        BT_TEMPO_DESCOBERTA);
                startActivityForResult(discoverableIntent, BT_VISIVEL);
                break;
        }
        return super.onOptionsItemSelected(item);
    }

    public void onCancel(DialogInterface dialog) {
        mAdaptadorBluetooth.cancelDiscovery();
        paraTudo();
    }

    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == BT_ATIVAR) {
            if (RESULT_OK != resultCode) {
                Toast.makeText(this, R.string.msg_ativar_bluetooth, Toast.LENGTH_SHORT).show();
                finish();
            }
        } else if (requestCode == BT_VISIVEL) {
            if (resultCode == BT_TEMPO_DESCOBERTA) {
                iniciaThreadServidor();
            } else {
                Toast.makeText(this, R.string.msg_aparelho_invisivel, Toast.LENGTH_SHORT).show();
            }
        }
    }

    private void exibirDispositivosEncontrados() {
        mAguardeDialog.dismiss();

        String[] aparelhos = new String[mDispositivosEncontrados.size()];
        for (int i = 0; i < mDispositivosEncontrados.size(); i++) {
            aparelhos[i] = mDispositivosEncontrados.get(i).getName();
        }

        AlertDialog dialog = new AlertDialog.Builder(this)
                .setTitle(R.string.aparelhos_encontrados)
                .setSingleChoiceItems(aparelhos, -1, this)
                .create();
        dialog.show();
```

```
    }

    public void onClick(DialogInterface dialog, int which) {
        iniciaThreadCliente(which);
        dialog.dismiss();
    }

    public void onClick(View v) {
        EditText edt = (EditText) findViewById(R.id.edtMsg);
        String msg = edt.getText().toString();
        edt.setText("");
        try {
            if (os != null) {
                os.writeUTF(msg); //Escreve uma string no OutputStream
                mMensagens.add("Eu: " + msg);
                mMensagens.notifyDataSetChanged(); //atualiza a lista que exibe as mensagens
            }
        } catch (IOException e) {
            e.printStackTrace();
            mTelaHandler.obtainMessage(MSG_DESCONECTOU, e.getMessage() + "[0]").sendToTarget();
        }
    }

    private void exibirProgressDialog(int mensagem, long tempo) {
        mAguardeDialog = ProgressDialog.show(this, getString(R.string.aguarde),
                getString(mensagem), true, true, this);
        mAguardeDialog.show();
        if (tempo > 0) {
            mTelaHandler.postDelayed(new Runnable() { //agenda a execução de uma ação
                public void run() {                   //após um determinado tempo
                    if (mThreadComunicacao == null) {
                        mAguardeDialog.cancel();
                    }
                }
            }, tempo * 1000);
        }
    }

    private void paraTudo() {
        if (mThreadComunicacao != null) {
            mThreadComunicacao.parar();
            mThreadComunicacao = null;
        }
        if (mThreadServidor != null) {
            mThreadServidor.parar();
            mThreadServidor = null;
        }
        if (mThreadCliente != null) {
            mThreadCliente.parar();
            mThreadCliente = null;
        }
    }

    private void iniciaThreadServidor() {
        exibirProgressDialog(R.string.mensagem_servidor, BT_TEMPO_DESCOBERTA);
        paraTudo();

        mThreadServidor = new ThreadServidor();
        mThreadServidor.iniciar();
    }

    private void iniciaThreadCliente(final int which) {
        paraTudo();
        mThreadCliente = new ThreadCliente();
        mThreadCliente.iniciar(mDispositivosEncontrados.get(which));
    }

    private void trataSocket(final BluetoothSocket socket) {
        mAguardeDialog.dismiss();

        mThreadComunicacao = new ThreadComunicacao();
        mThreadComunicacao.iniciar(socket);
    }

    private class ThreadServidor extends Thread {

        BluetoothServerSocket serverSocket;
        BluetoothSocket clientSocket;

        public void run() {
            try {
                serverSocket = mAdaptadorBluetooth.
                        listenUsingRfcommWithServiceRecord(SERVICO, MEU_UUID);
                clientSocket = serverSocket.accept();
```

```
                trataSocket(clientSocket);
            } catch (IOException e) {
                mTelaHandler.obtainMessage(MSG_DESCONECTOU, e.getMessage() + "[1]").sendToTarget();
                e.printStackTrace();
            }
        }

        public void iniciar(){
            start();
        }

        public void parar(){
            try {
                serverSocket.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    private class ThreadCliente extends Thread {
        BluetoothDevice device;
        BluetoothSocket socket;

        public void run() {
            try {
                BluetoothSocket socket = device.createRfcommSocketToServiceRecord(MEU_UUID);
                socket.connect();
                trataSocket(socket);
            } catch (IOException e) {
                e.printStackTrace();
                mTelaHandler.obtainMessage(MSG_DESCONECTOU, e.getMessage() + "[2]").sendToTarget();
            }
        }

        public void iniciar(BluetoothDevice device){
            this.device = device;
            start();
        }

        public void parar(){
            try {
                socket.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }

    private class ThreadComunicacao extends Thread {
        String nome;
        BluetoothSocket socket;

        public void run() {
            try {
                nome = socket.getRemoteDevice().getName(); //Busca o nome do dispositivo remoto
                is = new DataInputStream(socket.getInputStream());
                os =new DataOutputStream(socket.getOutputStream());
                String string;
                while (true) {
                    string = is.readUTF();
                    mTelaHandler.obtainMessage(MSG_TEXTO, nome + ": " + string).sendToTarget();
                }
            } catch (IOException e) {
                e.printStackTrace();
                mTelaHandler.obtainMessage(MSG_DESCONECTOU,
                        e.getMessage() + "[3]").sendToTarget();
            }
        }
        public void iniciar(BluetoothSocket socket){
            this.socket = socket;
            start();
        }
        public void parar(){
            try {
                is.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
            try {
                os.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
```

```
        }
    }

    private class EventosBluetoothReceiver extends BroadcastReceiver {

        public void onReceive(Context context, Intent intent) {
            if (BluetoothDevice.ACTION_FOUND.equals(intent.getAction())){
                BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
                mDispositivosEncontrados.add(device);

            } else if (BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(intent.getAction())){
                exibirDispositivosEncontrados();
            }
        }
    }

    private class TelaHandler extends Handler {

        public void handleMessage(Message msg) {
            super.handleMessage(msg);

            if (msg.what == MSG_TEXTO){
                mMensagens.add(msg.obj.toString());
                mMensagens.notifyDataSetChanged();

            } else if (msg.what == MSG_DESCONECTOU){
                Toast.makeText(BluetoothChatActivity.this,
                        getString(R.string.msg_desconectou) + msg.obj.toString(),
                        Toast.LENGTH_SHORT).show();
            }
        }
    }

}
```