Correction TP09

Matière : ATELIER DEVELOPPEMENT MOBILE | **Classes : SEM31**

Exercice1 : Bras

Serveur : Bras

```
package bras;
public class GlgRobotArmDemo extends GlgJBean implements ActionListener {
        private static JSlider[] tSlider;
    private static int indice;
    private static final int PORT = 6035;
    private ServerSocket ss;
    private Socket s;
    private BufferedReader br;
    public GlgRobotArmDemo() {
        super();
        SetDResource("$config/GlgAntiAliasing", 1.0); // Enable antialiasing
        lancerThreadServeur();
    }
    private void lancerThreadServeur() {
        Runnable r = new Runnable() {
            @Override
            public void run() {
                demmarerServeur();
            }
        };
        Thread th = new Thread(r);
        th.start();
    }
    private void demmarerServeur() {
        try {
            ss = new ServerSocket(PORT);
            s = ss.accept();
            br = new BufferedReader(new InputStreamReader(s.getInputStream()));
            while (true) {
                String cmd;
                cmd = br.readLine();
                executerCommande(cmd);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    private void executerCommande(String cmd) {
        System.out.println(cmd);
        String[] t = cmd.split(":");
        if (t.length > 1) {
            int indice = Integer.parseInt(t[0]);
            int val = Integer.parseInt(t[1]);
            tSlider[indice].setValue(val);
        }
    }
}
```

Client : ControleurBras

```
package com.bras;
public class MainActivity extends Activity {
    private SeekBar[] tSeek;
    private Button btnConnecter;
    public static final int PORT_SERVEUR = 6035;
    public static final String ADR_SERVEUR = "10.0.2.2";
    private Socket s;
    private PrintWriter pw;
    private EditText edAdresse;
    private EditText edPort;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        init();
    }
    private void init() {
        tSeek = new SeekBar[7];
        tSeek[6] = (SeekBar) findViewById(R.id.seekBar);
        tSeek[5] = (SeekBar) findViewById(R.id.seekBar1);
        tSeek[4] = (SeekBar) findViewById(R.id.seekBar2);
        tSeek[3] = (SeekBar) findViewById(R.id.seekBar3);
        tSeek[2] = (SeekBar) findViewById(R.id.seekBar4);
        tSeek[1] = (SeekBar) findViewById(R.id.seekBar5);
        tSeek[0] = (SeekBar) findViewById(R.id.seekBar6);
        btnConnecter = (Button) findViewById(R.id.btnConnecter);
        edAdresse = (EditText) findViewById(R.id.edAdresse);
        edPort = (EditText) findViewById(R.id.edPort);
        edPort.setText(PORT_SERVEUR + "");
        edAdresse.setText(ADR_SERVEUR + "");
        ajouterEcoteur();
    }
    private void ajouterEcoteur() {
        btnConnecter.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                lanceThreadClient();
            }
        });
        for (int j = 0; j < tSeek.length; j++) {
            final int finalJ = j;
            tSeek[j].setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
                @Override
                public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
                    envoyer(finalJ + ":" +  seekBar.getProgress());
                }

                @Override
                public void onStartTrackingTouch(SeekBar seekBar) {

                }

                @Override
                public void onStopTrackingTouch(SeekBar seekBar) {

                }
            });
        }
}
```

```
    private void lanceThreadClient() {
        Runnable r = new Runnable() {
            @Override
            public void run() {
                demmarerClient();
            }

        };
        Thread th = new Thread(r);
        th.start();
        btnConnecter.setVisibility(View.GONE);
    }

    private void demmarerClient() {
        try {
            Inet4Address i = (Inet4Address) Inet4Address.getByName(edAdresse.getText().toString());
            s = new Socket(i, Integer.parseInt(edPort.getText().toString()));
            pw = new PrintWriter(new BufferedWriter(new OutputStreamWriter(s.getOutputStream())), true);
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    private void envoyer(final String cmd) {
        Runnable r = new Runnable() {
            @Override
            public void run() {
                if (pw != null) {
                    pw.println(cmd);
                    pw.flush();
                }
            }
        };
        Thread th = new Thread(r);
        th.start();

    }
}
```