NFC Secure Element Example

Warning: This example was created for the NFC implementation provided on Series 40 devices, and may include use of features not yet available for NFC on Symbian devices.

The code shows a basic MIDlet that communicates with the internal secure element.

In order to run this example you also need the appropriate Java Card Example (For Emulator/Real Phone).

In order to run this on the phone, you do have to unlock it and you need as well a code signing certificate to sign this MIDlet. In the emulator you have to set the security domain (Tools => Preferences) to "MAXIMUM". (Shown at NFC Forum Developer Training, WIMA 2008 and NFC-Congress Hagenberg 2008) Download appropriate NetBeans Project: File:Ticket.zip.

```
package at.nfcresearch.wima.examples;

import java.io.IOException;
import javax.microedition.contactless.ContactlessException;
import javax.microedition.contactless.sc.ISO14443Connection;
import javax.microedition.io.Connector;
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class Ticketing extends MIDlet implements CommandListener {

    // APDUS in order to communicate with the JavaCard Applet
    private byte[] SELECT = {(byte) 0x00, (byte) 0xA4, (byte) 0x04, (byte) 0x00, (byte)
0x09, (byte) 0x74, (byte) 0x69, (byte) 0x63, (byte) 0x6B, (byte) 0x65, (byte) 0x74,
(byte) 0x69, (byte) 0x6E, (byte) 0x67, (byte) 0x00};
    private byte[] INS_INC = {(byte) 0x00, (byte) 0x01, (byte) 0x00, (byte) 0x00};
    private byte[] INS_DEC = {(byte) 0x00, (byte) 0x02, (byte) 0x00, (byte) 0x00};
    private byte[] INS_READ = {(byte) 0x00, (byte) 0x03, (byte) 0x00, (byte) 0x00,
(byte) 0x01};
    // UI Stuff
    private Command exitCommand;
    private Command incCommand;
    private Command decCommand;
    private Form form;
    private StringItem info;
    private StringItem value;
    // Connetivity with Secure Element
    private String uri;
    private ISO14443Connection conn;

    public Ticketing() {
        exitCommand = new Command("Exit", Command.EXIT, 1);
        incCommand = new Command("INC", Command.ITEM, 2);
        decCommand = new Command("DEC", Command.ITEM, 3);
        info = new StringItem("Info:", "-- just started --");
        value = new StringItem("Value:", "-- just started --");
        form = new Form("NFC-Research: Secure Element Demo");

        form.append(info);
        form.append(value);
        form.addCommand(exitCommand);
        form.addCommand(incCommand);
        form.addCommand(decCommand);
```


(C) Copyright Nokia 2014. All rights reserved.

```
        form.setCommandListener(this);

        try {

            // get URI of the secure element from the system
            uri = System.getProperty("internal.se.url");

            // Opening the Conneciton to the Secure Element
            conn = (ISO14443Connection) Connector.open(uri);

            // Selecting the Applet
            byte[] result = conn.exchangeData(SELECT);

            // Check if select was oaky
            if (result[0] == (byte) 0x90 && result[1] == (byte) 0x00) {
                info.setText("Select okay");
            }

            readValue();

        } catch (IOException ie) {
            info.setText("Could not Select: " + ie.toString());
        } catch (ContactlessException ce) {
            info.setText("Error on Select: " + ce.toString());
        } catch (SecurityException se) {
            info.setText("Error, Application not trusted (sign MIDLet!): " +
se.toString());
        }


    }

    public void startApp() {

        Display.getDisplay(this).setCurrent(form);

    }

    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {
    }

    public void commandAction(Command command, Displayable displayable) {
        if (command == exitCommand) {
            try {
                if (conn != null) {
                    conn.close();
                }

            } catch (IOException ex) {
                ex.printStackTrace();
            }
            destroyApp(false);
            notifyDestroyed();
        }
```


(C) Copyright Nokia 2014. All rights reserved.

```
        // Increase Value in Secure Element
        if (command == incCommand) {
            if (incValue()) {
                readValue();
            }

        }
        // decrease Value in Secure Element
        if (command == decCommand) {
            if (decValue()) {
                readValue();
            }
        }
    }

    /**
     * Method for increasing the value in the JavaCard Applet
     */
    private boolean incValue() {
        byte[] result = null;
        try {
            result = conn.exchangeData(INS_INC);
        } catch (IOException io) {
            displayAlert("IO-Error during Increasing value: " + io.toString(),
AlertType.ERROR);
            return false;
        } catch (ContactlessException ce) {
            displayAlert("CL-Error during Increasing value: " + ce.toString(),
AlertType.ERROR);
            return false;
        }

// check if read was okay
        if (result.length == 2 && result[0] == (byte) 0x90 && result[1] == (byte) 0x00)
{
            info.setText("INC okay");
            value.setText("" + result[0]);
            return true;
        } else {
            displayAlert("Error Increasing Value", AlertType.ERROR);
            info.setText("ERROR while Increasing!");
            value.setText("n/a");
            return false;
        }
    }

    /**
     * Method for decreasing the value in the JavaCard Applet
     */
    private boolean decValue() {
        byte[] result = null;
        try {
            result = conn.exchangeData(INS_DEC);
        } catch (IOException io) {
            displayAlert("IO-Error during Reading value: " + io.toString(),
```


(C) Copyright Nokia 2014. All rights reserved.

```
AlertType.ERROR);
            return false;
        } catch (ContactlessException ce) {
            displayAlert("CL-Error during Reading value: " + ce.toString(),
AlertType.ERROR);
            return false;
        }

        if (result.length == 2 && result[0] == (byte) 0x90 && result[1] == (byte) 0x00)
{
            info.setText("decrease okay");
            value.setText("" + result[0]);
            return true;
        } else {
            displayAlert("Error Decreasing Value", AlertType.ERROR);
            info.setText("ERROR while Decreasing!");
            value.setText("n/a");
            return false;
        }
    }

    /**
     * Method for reading the value in the JavaCard Applet
     */
    private void readValue() {
        // Read value again

        byte[] result = null;
        try {
            result = conn.exchangeData(INS_READ);
        } catch (IOException io) {
            displayAlert("IO-Error during Reading value: " + io.toString(),
AlertType.ERROR);
        } catch (ContactlessException ce) {
            displayAlert("CL-Error during Reading value: " + ce.toString(),
AlertType.ERROR);
        }

// check if read was okay
        if (result.length == 3 && result[1] == (byte) 0x90 && result[2] == (byte) 0x00)
{
            info.setText("read okay");
            value.setText("" + result[0]);
        } else {
            displayAlert("Error Reading Value", AlertType.ERROR);
            info.setText("ERROR while Reading!");
            value.setText("n/a");
        }

    }

    private void displayAlert(String error, AlertType type) {
        Alert err = new Alert(form.getTitle(), error, null, type);
        Display.getDisplay(this).setCurrent(err, form);
    }
}
```


(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.