Como criar uma URL usando NFC

Este código mostra um MIDlet básico que escreve uma URL para uma tag/Transponder externo. (Mostrado no NFC Forum Developer Training, WIMA 2008 e NFC-Congress Hagenberg 2008) Baixe o apropriado NetBeans projetoDownload: File:URLWriter.zip.

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

// Packages for contactless Communcation
import java.io.IOException;
import javax.microedition.contactless.ContactlessException;
import javax.microedition.contactless.DiscoveryManager;
import javax.microedition.contactless.TargetProperties;
import javax.microedition.contactless.TargetListener;
import javax.microedition.contactless.TargetType;

// Packages for NDEF
import javax.microedition.contactless.ndef.NDEFRecord;
import javax.microedition.contactless.ndef.NDEFMessage;
import javax.microedition.contactless.ndef.NDEFRecordType;
import javax.microedition.contactless.ndef.NDEFTagConnection;
import javax.microedition.io.Connector;


// Packages for GUI Stuff
import javax.microedition.lcdui.Alert;
import javax.microedition.lcdui.AlertType;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.StringItem;

import javax.microedition.midlet.*;

public class URLWriter extends MIDlet implements TargetListener, CommandListener {

    // Display Stuff
    private Command exitCommand;
    private Form form;

    public URLWriter() {
        exitCommand = new Command("Exit", Command.EXIT, 1);
        form = new Form("NFC-Research.at: URL Writer");

        form.addCommand(exitCommand);
        form.setCommandListener(this);
        form.append("Touch Tag to write URL");

        // Registration of the TargetListener for external contactless
        // Targets (in this RFID_TAG).
        try {
            DiscoveryManager dm = DiscoveryManager.getInstance();
            dm.addTargetListener(this, TargetType.NDEF_TAG);

        } catch (ContactlessException ce) {
            displayAlert("Unable to register TargetListener: " + ce.toString(), AlertType.ERROR);
        }

    }

    public void startApp() {
        Display.getDisplay(this).setCurrent(form);
    }

    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {

    }

    /**
     * Implementation of the Call-Back Function of the TargetListener
     * @param targetProperties: Array of Targets found by the Phone
     *
     */
    public void targetDetected(TargetProperties[] targetProperties) {

        // in case no targets found, exit the method
        if (targetProperties.length == 0) {
            return;
        }

        // NDEF Connection for write Operation
        NDEFTagConnection ndconn = getNDEFTAGConnection(targetProperties);
        try {
            // Open NDEF Connection to the first Tag found
            // ndconn = (NDEFTagConnection) Connector.open(targetProperties[0].getUrl());


            // Headerbyte: open Bookmark in Browser
            byte[] headerByte = {0x00};

            // Payload Text: URL as a byte-Array
            byte[] urlBytes = "http://www.nfc-research.com".getBytes();

            // Create NDEF Record
            NDEFRecord r = new NDEFRecord(new NDEFRecordType(NDEFRecordType.NFC_FORUM_RTD, "urn:nfc:wkt:U"), null, null);

            // Append Payload Manually
            r.appendPayload(headerByte);
            r.appendPayload(urlBytes);

            // Create Recoard Array and Add Record
            NDEFRecord[] myRecArray = new NDEFRecord[]{r};

            // Generate a Message out of the Array
            NDEFMessage myMessage = new NDEFMessage(myRecArray);
```


(C) Copyright Nokia 2013. All rights reserved.

```
            // Write Message to Tag
            ndconn.writeNDEF(myMessage);


            displayAlert("Information written", AlertType.INFO);

        } catch (IOException io) {
            displayAlert("IOException during open() or Write(): " + io.toString(), AlertType.ERROR);
        } catch (ContactlessException io) {
            displayAlert("ContactlessException during writeNDEF(): " + io.toString(), AlertType.ERROR);
        } catch (Exception e) {
            displayAlert("Exception: " + e.toString(), AlertType.ERROR);
        } // in case of an exception, close connection properly
        finally {

            // close Connection again
            if (ndconn != null) {
                try {
                    ndconn.close();
                } catch (IOException ex) {
                    displayAlert("IOException during close(): " + ex.toString(), AlertType.ERROR);
                }
            }
        }

    }

    private NDEFTagConnection getNDEFTAGConnection(TargetProperties[] tProp) {
        for (int j = 0; j < tProp.length; j++) {
            Class[] connections = tProp[j].getConnectionNames();
            if (connections != null) {
                for (int i = 0; i < connections.length; i++) {
                    if (connections[i].getName().equals(
                            "javax.microedition.contactless.ndef.NDEFTagConnection")) {
                        try {
                            return (NDEFTagConnection) Connector.open(tProp[0].getUrl(connections[i]));
                        } catch (Exception e) {
                            displayAlert("Error writting NDEF: " + e.toString(), AlertType.ERROR);
                        }
                    }
                }
            }
        }
        return null;
    }

    /**
     * Implementation of the Call-Back function of the CommandListener
     * @param command: command key pressed
     * @param displayable: associated displayable Object
     */
    public void commandAction(Command command, Displayable displayable) {
        if (command == exitCommand) {
            DiscoveryManager dm = DiscoveryManager.getInstance();
            dm.removeTargetListener(this, TargetType.NDEF_TAG);
            destroyApp(false);
            notifyDestroyed();
        }
    }

    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 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.