NFC Interactor Example

This example uses the Near Field Communication (NFC) JSR 257 from Java ME to read, write and delete NDEF-type tags. It's partly based on and extends the NFC URLWriter example. Download the latest version of the NFC Interactor from here.

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.

Features

- Reads NDEF tags.
- Distinguishes between Smart Posters, pure URI records and text records; shows the overview info of other tag types.
- Writes Smart Poster tags. URL, title, action and icon can be defined in the UI.
- Writes URI records, automatically encoded with all defined URI abbreviations.
- Writes Text records; the default English language / UTF-8 encoding can be adapted in the source code.
- Writes MIME records for PNG images; the image can be switched in the source code.
- When writing, it automatically shortens the URL tag for all defined prefixes.
- Deletes messages on a tag by writing an empty message.
- Handles tag processing in a separate thread, as recommended by the API docs.
- Complete source code documentation.

Limitations

- When reading, it only shows the contents of the message, but doesn't parse them (e.g., to extract the URL and the title of the poster into extra variables).
- Only uses the default character encoding.
- The app is a flexible starting point for NDEF message experiments. For other use cases, you should extend the app to separate the MIDlet, the UI, the NFC interaction and the various NDEF message types into extra classes.

Reading tags

Note that the source code snippets rely on other components within the whole example app (e.g., for opening the connection to the tag). Download the full example from the bottom of the page to get the full project.

```
/**
 * Processing method to read an NDEF message from a pre-established tag connection.
 * @param ndconn established connection to the NDEF tag.
 * @throws IOException
 * @throws ContactlessException
 */
private void readNDEFMessage(NDEFTagConnection ndconn) throws IOException,
ContactlessException {
    // Get the message out of the connection
    NDEFMessage ndefMessage = ndconn.readNDEF();
    if (ndefMessage == null || ndefMessage.getNumberOfRecords() <= 0) {
        // No records are found, or no message contained in the connection
        displayAlert("Read NDEF", "No records in the message.", AlertType.ERROR);
    } else {
        boolean recordHandled;
        // Go through all the records present in the message
        for (int i = 0; i < ndefMessage.getNumberOfRecords(); i++) {
            recordHandled = false;
            NDEFRecord rec = ndefMessage.getRecord(i);
            NDEFRecordType type = rec.getRecordType();
            String typeName = type.getName();
```


(C) Copyright Nokia 2014. All rights reserved.

```
            if (typeName != null) {
                // Enable specific tag handling
                if (type.getName().equals("Sp")) {
                    // Show Smart Poster
                    // Doesn't do actual parsing of the contents
                    byte[] payload = rec.getPayload();
                    displayAlert("Smart Poster", new String(payload),
AlertType.CONFIRMATION);
                    recordHandled = true;

                } else if (type.getName().equals("U")) {
                    // Parse URL
                    // Doesn't do actual parsing of the contents
                    byte[] payload = rec.getPayload();
                    displayAlert("Url", new String(payload), AlertType.CONFIRMATION);
                    recordHandled = true;
                }
            }
            if (!recordHandled) {
                // Didn't do special parsing of the record - so just show general info
about it
                displayAlert("Record " + (i + 1) + "/" +
ndefMessage.getNumberOfRecords(), "format = " + type.getFormat() + ", name = " +
type.getName() + "\n", AlertType.CONFIRMATION);
            }
        }
    }
}
```

Writing tags

```
/**
 * Processing method to write a Smart Poster NDEF message to a pre-established tag
connection.
 * Will take the title and the URL from the UI elements in the form.
 * @param ndconn established connection to the NDEF tag.
 * @throws IOException
 * @throws ContactlessException
 */
private void writeNDEFMessage(NDEFTagConnection ndconn) throws IOException,
ContactlessException {

    // Create NDEFMessage
    NDEFMessage message = new NDEFMessage();

    // Compile the URL
    byte[] urlPrefix = new byte[1];
    byte[] url;
    String fullUrl = targetUrl.getString();
    // There are many more abbreviations possible - here, just handle
    // the most common case with "http://www." and reduce this text
    // to one byte, instead of storing the whole string.
    if (fullUrl.startsWith("http://www.")) {
        // Prefix is 1, which will tell the reader to prepend "http://www."
        // to the URL
```


(C) Copyright Nokia 2014. All rights reserved.

```
        urlPrefix[0] = (byte) 0x01;
        // Cut the "http://www." part away from the URL
        url = fullUrl.substring(11, fullUrl.length()).getBytes();
    } else {
        // Store the full URL to the tag
        urlPrefix[0] = (byte) 0x00;
        url = fullUrl.getBytes();
    }

    // Create the record for the URL
    NDEFRecord recordUrl = new NDEFRecord(new
NDEFRecordType(NDEFRecordType.NFC_FORUM_RTD, "urn:nfc:wkt:U"), null, null);
    recordUrl.appendPayload(urlPrefix);
    recordUrl.appendPayload(url);

    // Append the record to the message
    message.appendRecord(recordUrl);

    // Title
    byte[] lang = "en-US".getBytes("US-ASCII");
    byte[] lang_len = {(byte) lang.length};
    byte[] title = posterTitle.getString().getBytes("utf-8");

    NDEFRecord recordTitle = new NDEFRecord(new
NDEFRecordType(NDEFRecordType.NFC_FORUM_RTD, "urn:nfc:wkt:T"), null, null);
    recordTitle.appendPayload(lang_len);
    recordTitle.appendPayload(lang);
    recordTitle.appendPayload(title);

    // Append the record to the message
    message.appendRecord(recordTitle);

    // Create the Smart Poster record
    NDEFRecord recordSmartPoster = new NDEFRecord(new
NDEFRecordType(NDEFRecordType.NFC_FORUM_RTD, "urn:nfc:wkt:Sp"), null, null);
    // Add the URL and the title as a payload to the Smart Poster record.
    recordSmartPoster.appendPayload(message.toByteArray()); // add content of previously
created NDEFMessage

    // Create the final Smart Poster meta-message
    NDEFMessage messageSmartPoster = new NDEFMessage();

    // Append the smart poster record to the meta-message
    messageSmartPoster.appendRecord(recordSmartPoster);

    // Write message to the tag
    ndconn.writeNDEF(messageSmartPoster);

    displayAlert(form.getTitle(), "Information written", AlertType.CONFIRMATION);
}
```

Deleting tags

```
/**
 * Processing method to delete an NDEF message from a pre-established tag connection.
```


(C) Copyright Nokia 2014. All rights reserved.

```
 * Writes an empty message if a message already exists on the tag. Doesn't
 * do anything if the tag is already empty.
 * @param ndconn established connection to the NDEF tag.
 * @throws ContactlessException
 * @throws IOException
 */
private void deleteNDEFMessage(NDEFTagConnection ndconn) throws ContactlessException,
IOException {
    NDEFMessage ndefMessage = ndconn.readNDEF();
    if (ndefMessage == null || ndefMessage.getNumberOfRecords() <= 0) {
        displayAlert("Delete NDEF", "Tag already empty.", AlertType.ERROR);
    } else {
        NDEFRecord emptyRecord = new NDEFRecord(new NDEFRecordType(NDEFRecordType.EMPTY,
null), null, null);
        NDEFRecord[] emptyRecordArray = {emptyRecord};
        NDEFMessage emptyMessage = new NDEFMessage(emptyRecordArray);
        ndconn.writeNDEF(emptyMessage);
        displayAlert("Delete NDEF", "Wrote empty message.", AlertType.CONFIRMATION);
    }
}
```

Test application

The sample project contains the executable (.jar) that can be deployed to an NFC capable phone, plus the full source code of a NetBeans Java ME project. Download it from here

(C) Copyright Nokia 2014. All rights reserved.