Archived:Introduction to NFC on Nokia 6131 NFC and Nokia 6212 classic

Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

This article applies to a significantly older version of Series 40.

This article needs to be updated: If you found this article useful, please fix the problems below then delete the {{ArticleNeedsUpdate}} template from the article to remove this warning.

Reasons:
This article is relatively old, and is likely to be out of date in some aspects. The useful content in this article should be merged with the "official" About NFC overview and it should then be deleted.

Near Field Communication (or NFC) is a short range wireless connectivity technology that is activated when NFC enabled devices or tags are brought within touching distance. It is a relatively young technology that is gaining wide use in a number of applications including mobile ticketing, smart cards and Bluetooth pairing, to name just a few.

This article provides a brief overview of NFC, and NFC on Nokia platforms. Note that About NFC is a more recent article, and is a better starting point.

Working Method

NFC basically work in four simple ways:

1. Phone to Tag
2. Phone to Phone
3. Phone to Device
4. Phone to Reader

Phone to Tag

In this kind of application or development there should be a tag in poster or any other object and when we put our phone near that object it initiates the communication and read that tag from that object. The basic example of this kind of application is as follows:

1. call will start when user touches the mobile to the NFC card.
2. alarm will be set when user touches the mobile to NFC card.

Phone to Phone

In this kind of application two phones communicate with each other and transfer or share objects with each other. The ideal example can be one NFC phone shares a picture with another NFC mobile by touching each other.

Phone to Device

In this kind of application the phone will communicate with any device. For example using NFC we can connect Bluetooth head set with phone by touching or placing them close to each other.

Phone to Reader

In this kind of application the phone will communicate with any external reader. Suppose reader can be available in any transportation company and people can communicate with that reader and purchase their ticket.

 (C) Copyright Nokia 2014. All rights reserved.

The Contactless Communication API

The Contactless Communication API (JSR 257) is used in development for NFC. The Contactless Communication API allows you to discover and exchange data with contactless targets such as NDEF tags, RFID tags, and external smartcards. The API also provides support for visual tags. It has the following packages.

1. javax.microedition.contactless (Mandatory)
   - target discovery and classes common to all targets
2. javax.microedition.contactless.ndef
   - communicate with tags (NDEF formatted data)
3. javax.microedition.contactless.rf
   - communicate with RFID (no NDEF formatted data)
4. javax.microedition.contactless.sc
   - communication with external smart cards
5. javax.microedition.contactless.visual
   - reading and generating visual tags

Register Target

First you need to register for Target by following code snippts public void registerTargetListeners(TargetListener targetListener) { // Discover supported types

Invalid language.

You need to specify a language like this: `<source lang="html4strict">...</source>`

Supported languages for syntax highlighting:

`4cs`, `6502acme`, `6502kickass`, `6502tasm`, `68000devpac`, `abap`, `actionscript`, `actionscript3`, `ada`, `algol68`, `apache`, `applescript`, `apt_sources`, `asm`, `asp`, `autoconf`, `autohotkey`, `autoit`, `avisynth`, `awk`, `bascomavr`, `bash`, `basic4gl`, `bf`, `bibtex`, `blitzbasic`, `bnf`, `boo`, `c`, `c_loadrunner`, `c_mac`, `caddcl`, `cadlisp`, `cfdg`, `cfm`, `chaiscript`, `cil`, `clojure`, `cmake`, `cobol`, `coffeescript`, `cpp`, `cpp-qt`, `csharp`, `css`, `cuesheet`, `d`, `dcs`, `delphi`, `diff`, `div`, `dos`, `dot`, `e`, `ecmascript`, `eiffel`, `email`, `epc`, `erlang`, `euphoria`, `f1`, `falcon`, `fo`, `fortran`, `freebasic`, `fsharp`, `gambas`, `gdb`, `genero`, `genie`, `gettext`, `glsl`, `gml`, `gnuplot`, `go`, `groovy`, `gwbasic`, `haskell`, `hicest`, `hq9plus`, `html4strict`, `html5`, `icon`, `idl`, `ini`, `inno`, `intercal`, `io`, `j`, `java`, `java5`, `javascript`, `jquery`, `kixtart`, `klonec`, `klonecpp`, `latex`, `lb`, `lisp`, `llvm`, `locobasic`, `logtalk`, `lolcode`, `lotusformulas`, `lotusscript`, `lscript`, `lsl2`, `lua`, `m68k`, `magiksf`, `make`, `mapbasic`, `matlab`, `mirc`, `mmix`, `modula2`, `modula3`, `mpasm`, `mxml`, `mysql`, `newlisp`, `nsis`, `oberon2`, `objc`, `objeck`, `ocaml`, `ocaml-brief`, `oobas`, `oracle11`, `oracle8`, `oxygene`, `oz`, `pascal`, `pcre`, `per`, `perl`, `perl6`, `pf`, `php`, `php-brief`,

 (C) Copyright Nokia 2014. All rights reserved.

pic16, pike, pixelbender, pli, plsql, postgresql, povray, powerbuilder, powershell, proftpd, progress, prolog, properties, providex, purebasic, pycon, python, q, qbasic, rails, rebol, reg, robots, rpmspec, rsplus, ruby, sas, scala, scheme, scilab, sdlbasic, smalltalk, smarty, sql, systemverilog, tcl, teraterm, text, thinbasic, tsql, typoscript, unicon, uscript, vala, vb, vbnet, verilog, vhdl, vim, visualfoxpro, visualprolog, whitespace, whois, winbatch, xbasic, xml, xorg_conf, xpp, yaml, z80, zxbasic

```
TargetType[] tp = DiscoveryManager.getSupportedTargetTypes();
try {
// Register listener for each of the supported types
for (int i=0; i<tp.length; i++) {
if (tp[i].equals((TargetType.ISO14443_CARD))) {
dm.addTargetListener(
targetListener, TargetType.ISO14443_CARD);
} else...
:
:
}
}
} catch (Exception e) {
// ...
}
}
```

Once the target will be registered it will invoke the `targetDetected()` whenever any target comes near it and we can get the NDEF records in it and we can do further communication.

Invalid language.

You need to specify a language like this: `<source lang="html4strict">...</source>`

Supported languages for syntax highlighting:

4cs, 6502acme, 6502kickass, 6502tasm, 68000devpac, abap, actionscript, actionscript3, ada, algol68, apache, applescript, apt_sources, asm, asp, autoconf, autohotkey, autoit, avisynth, awk, bascomavr, bash, basic4gl, bf, bibtex, blitzbasic, bnf, boo, c, c_loadrunner, c_mac, caddcl, cadlisp, cfdg, cfm, chaiscript, cil, clojure, cmake, cobol, coffeescript, cpp, cpp-qt, csharp, css, cuesheet, d, dcs, delphi, diff, div, dos, dot, e, ecmascript, eiffel, email, epc, erlang, euphoria, f1, falcon, fo, fortran, freebasic, fsharp, gambas, gdb, genero, genie, gettext, glsl, gml, gnuplot, go, groovy, gwbasic, haskell, hicest, hq9plus, html4strict, html5, icon, idl, ini, inno, intercal, io, j, java, java5, javascript, jquery, kixtart, klonec, klonecpp, latex, lb, lisp, llvm, locobasic, logtalk, lolcode, lotusformulas, lotusscript, lscript, lsl2, lua, m68k, magiksf, make, mapbasic, matlab, mirc, mmix, modula2, modula3, mpasm, mxml, mysql, newlisp, nsis, oberon2, objc, objeck, ocaml, ocaml-brief, oobas, oracle11, oracle8, oxygene, oz, pascal, pcre, per, perl, perl6, pf, php, php-brief, pic16, pike, pixelbender, pli, plsql, postgresql, povray, powerbuilder, powershell, proftpd, progress, prolog, properties, providex, purebasic, pycon, python, q, qbasic, rails, rebol, reg, robots, rpmspec, rsplus, ruby, sas, scala, scheme, scilab, sdlbasic, smalltalk, smarty, sql, systemverilog, tcl, teraterm, text, thinbasic, tsql, typoscript, unicon, uscript, vala, vb, vbnet, verilog, vhdl, vim, visualfoxpro, visualprolog, whitespace, whois, winbatch, xbasic, xml, xorg_conf, xpp, yaml, z80, zxbasic

```
public void targetDetected(TargetProperties[] prop) {
try {
```

 (C) Copyright Nokia 2014. All rights reserved.

```
// Select the first target
TargetProperties target = prop[0];
// Get URL
String url = target.getUrl();
// Open NDEFTagConnection to the target
NDEFTagConnection conn = (
NDEFTagConnection) Connector.open(url);
// Read Message
NDEFMessage m = conn.readNDEF();
NDEFRecord[] r = m.getRecords();
// Process message / records, modify record for writing out
:
// Write message out, close connectin
conn.writeNDEF(m);
conn.close();
} catch (Exception e) {
}
}
```

NDEF Targets

In this section we will Register `NDEF_TAG` target (smart poster) to discover.

Invalid language.

You need to specify a language like this: `<source lang="html4strict">...</source>`

Supported languages for syntax highlighting:

`4cs`, `6502acme`, `6502kickass`, `6502tasm`, `68000devpac`, `abap`, `actionscript`, `actionscript3`, `ada`, `algol68`, `apache`, `applescript`, `apt_sources`, `asm`, `asp`, `autoconf`, `autohotkey`, `autoit`, `avisynth`, `awk`, `bascomavr`, `bash`, `basic4gl`, `bf`, `bibtex`, `blitzbasic`, `bnf`, `boo`, `c`, `c_loadrunner`, `c_mac`, `caddcl`, `cadlisp`, `cfdg`, `cfm`, `chaiscript`, `cil`, `clojure`, `cmake`, `cobol`, `coffeescript`, `cpp`, `cpp-qt`, `csharp`, `css`, `cuesheet`, `d`, `dcs`, `delphi`, `diff`, `div`, `dos`, `dot`, `e`, `ecmascript`, `eiffel`, `email`, `epc`, `erlang`, `euphoria`, `f1`, `falcon`, `fo`, `fortran`, `freebasic`, `fsharp`, `gambas`, `gdb`, `genero`, `genie`, `gettext`, `glsl`, `gml`, `gnuplot`, `go`, `groovy`, `gwbasic`, `haskell`, `hicest`, `hq9plus`, `html4strict`, `html5`, `icon`, `idl`, `ini`, `inno`, `intercal`, `io`, `j`, `java`, `java5`, `javascript`, `jquery`, `kixtart`, `klonec`, `klonecpp`, `latex`, `lb`, `lisp`, `llvm`, `locobasic`, `logtalk`, `lolcode`, `lotusformulas`, `lotusscript`, `lscript`, `lsl2`, `lua`, `m68k`, `magiksf`, `make`, `mapbasic`, `matlab`, `mirc`, `mmix`, `modula2`, `modula3`, `mpasm`, `mxml`, `mysql`, `newlisp`, `nsis`, `oberon2`, `objc`, `objeck`, `ocaml`, `ocaml-brief`, `oobas`, `oracle11`, `oracle8`, `oxygene`, `oz`, `pascal`, `pcre`, `per`, `perl`, `perl6`, `pf`, `php`, `php-brief`, `pic16`, `pike`, `pixelbender`, `pli`, `plsql`, `postgresql`, `povray`, `powerbuilder`, `powershell`, `proftpd`, `progress`, `prolog`, `properties`, `providex`, `purebasic`, `pycon`, `python`, `q`, `qbasic`, `rails`, `rebol`, `reg`, `robots`, `rpmspec`, `rsplus`, `ruby`, `sas`, `scala`, `scheme`, `scilab`, `sdlbasic`, `smalltalk`, `smarty`, `sql`, `systemverilog`, `tcl`, `teraterm`, `text`, `thinbasic`, `tsql`, `typoscript`, `unicon`, `uscript`, `vala`, `vb`, `vbnet`, `verilog`, `vhdl`, `vim`, `visualfoxpro`, `visualprolog`, `whitespace`, `whois`, `winbatch`, `xbasic`, `xml`, `xorg_conf`, `xpp`, `yaml`, `z80`, `zxbasic`

```
try {
NDEFRecordType spNrt = new NDEFRecordType(
NDEFRecordType.NFC_FORUM_RTD, "urn:nfc:wkt:Sp");
dm.addNDEFRecordListener(this, spNrt);
} catch (IllegalStateException e) {
:
} catch (Exception e) {
:
}
```

 (C) Copyright Nokia 2014. All rights reserved.

Once we register it then when it comes close and discovers record the recordDeetected() will be invoked. We can handle it in following way:

Invalid language.

You need to specify a language like this: `<source lang="html4strict">...</source>`

Supported languages for syntax highlighting:

4cs, 6502acme, 6502kickass, 6502tasm, 68000devpac, abap, actionscript, actionscript3, ada, algol68, apache, applescript, apt_sources, asm, asp, autoconf, autohotkey, autoit, avisynth, awk, bascomavr, bash, basic4gl, bf, bibtex, blitzbasic, bnf, boo, c, c_loadrunner, c_mac, caddcl, cadlisp, cfdg, cfm, chaiscript, cil, clojure, cmake, cobol, coffeescript, cpp, cpp-qt, csharp, css, cuesheet, d, dcs, delphi, diff, div, dos, dot, e, ecmascript, eiffel, email, epc, erlang, euphoria, f1, falcon, fo, fortran, freebasic, fsharp, gambas, gdb, genero, genie, gettext, glsl, gml, gnuplot, go, groovy, gwbasic, haskell, hicest, hq9plus, html4strict, html5, icon, idl, ini, inno, intercal, io, j, java, java5, javascript, jquery, kixtart, klonec, klonecpp, latex, lb, lisp, llvm, locobasic, logtalk, lolcode, lotusformulas, lotusscript, lscript, lsl2, lua, m68k, magiksf, make, mapbasic, matlab, mirc, mmix, modula2, modula3, mpasm, mxml, mysql, newlisp, nsis, oberon2, objc, objeck, ocaml, ocaml-brief, oobas, oracle11, oracle8, oxygene, oz, pascal, pcre, per, perl, perl6, pf, php, php-brief, pic16, pike, pixelbender, pli, plsql, postgresql, povray, powerbuilder, powershell, proftpd, progress, prolog, properties, providex, purebasic, pycon, python, q, qbasic, rails, rebol, reg, robots, rpmspec, rsplus, ruby, sas, scala, scheme, scilab, sdlbasic, smalltalk, smarty, sql, systemverilog, tcl, teraterm, text, thinbasic, tsql, typoscript, unicon, uscript, vala, vb, vbnet, verilog, vhdl, vim, visualfoxpro, visualprolog, whitespace, whois, winbatch, xbasic, xml, xorg_conf, xpp, yaml, z80, zxbasic

```
public void recordDetected(NDEFMessage ndefMessage) {
// Get records and record types from NDEF Message
NDEFRecordType[] rTypes = ndefMessage.getRecordTypes();
NDEFRecord[] records = ndefMessage.getRecords();
for (int i=0; i<records.length; i++) {
// Handle data, based on type of NDEFMessage
NDEFRecordType t = recordTypes[i];
NDEFRecord r = records[i];
byte[] id = r.getId();
long len = r.getPayloadLength();
byte[] p = r.getPayload();
// Process the record
// ...
}
}
```

Push Registry

If you have worked on messaging or any alarm based application then you might know about push registry. It is used to start application automatically. The same concept can be applied here. If the incoming connection is coming and if we register push registry then our application can start automatically. We can do push registry registration statically or dynamically as we did in other application of messaging.

Sector and Track wise Operations

There are many situations in which we want to communicate with RFID card in sector or Track wise.

Reading data

 (C) Copyright Nokia 2014. All rights reserved.

To read data of particular sector and block the following code snippet will be used. The following method will return the data of particular sector and block.

```
public String readBlock(int sector,int block,String url)
    {
                byte[] buf = new byte[16];
                byte[] bKeyA = {(byte)0xff,(byte)0xff,(byte)0xff,(byte)0xff,(byte)0xff,
                (byte)0xff};

                    MFKey.KeyA keyA = new MFKey.KeyA(bKeyA);

                MFStandardConnection tconn = null;
                // Make connection
                try {
                tconn = (MFStandardConnection)Connector.open(url);

                // read meta_data sector
                MFSector sec1 = tconn.getSector(sector);
                MFBlock mblk = sec1.getBlock(block);
                mblk.read(keyA, buf, 0, 0, 16);
                String sd=new String(buf);

                tconn.close();
                return  sd;

        }
        catch (Exception ex)
        {
            resultForm.append(ex.toString());
                buf = null;

        }
return null;
}
```

Writing data

```
    public void writeBlock(int sector,int block,String url,String data)
    {
        byte[] bKeyA = {(byte)0xff,(byte)0xff,(byte)0xff,(byte)0xff,(byte)0xff,
(byte)0xff};

MFKey.KeyA keyA = new MFKey.KeyA(bKeyA);

MFStandardConnection tconn = null;
// Make connection
try {
        tconn = (MFStandardConnection)Connector.open(url);

        // read meta_data sector
        MFSector sec1 = tconn.getSector(sector);
        //frm.append("Sector 1 Blocks:"+ sec.getBlockCount());

        MFBlock mblk = sec1.getBlock(block);
        /*
```

 (C) Copyright Nokia 2014. All rights reserved.

```
        byte writeData[]=new byte[16];
         writeData=data.getBytes();
         */
        byte[] buffer=new byte[16];
        for(int i=0;i<buffer.length;i++)buffer[i]=0x00;
        /*2*/
        byte[] writeData=data.getBytes();
        /*3*/
        System.arraycopy(writeData, 0, buffer, 0, writeData.length);
        /*4*/
        mblk.write(keyA, buffer, 0, 16, 0);
        mblk.write(keyA, writeData, 0,writeData.length,0);


//frm.append("data written successfully");
//frm.append("Got block 0");
}
catch(Exception e)
{
 //   frm.append(e.toString());

    }
finally
{
    try
    {
        if(tconn!=null)
        tconn.close();
    }
    catch(Exception e)
    {
        //frm.append(e.toString());
    }
}
```

Critical Part

To communicate with RFID card we have to pass secure keys so data can be confidential and no other application can read that data but if once the key has been lost then that particular sector may be useless for you so please take care before doing any change in secure keys. I think the person with beginner knowledge should not try to do any thing with keys of card.

NFC enables Handsets

As has been said NFC is still a very young technology so there are very few devices which are available for NFC.

- Nokia 6212 classic
- Nokia 6131 NFC
- Nokia 3220 + NFC Shell[15]
- Samsung SGH-X700 NFC[16]
- Samsung D500E[17]
- SAGEM my700X Contactless[18]
- LG 600V contactless[19]
- Motorola L7 (SLVR)[20]
- Benq T80[21]

Development Tools

To do development in Java ME you will need the SDK provided by Nokia Developer for NFC enabled handsets like Nokia 6212 and Nokia 6131 which you can get from the respective device's specifications page. Once you install the SDK you can integrate it

 (C) Copyright Nokia 2014. All rights reserved.

with NetBeans or Eclipse easily. Documentation is also available:File:S40 N6212 NFC SDK v1 0 en-DS S40 N6212 NFC SDK v1 0 en.zip

Security

The communication range for NFC is so limited and there are keys for security in communication but yet there are chances of attacks by data modification, eavesdropping and relay attacks. So to guard from these kinds of attacks applications have to use higher-layer cryptographic protocols (e.g., SSL) to establish a secure channel.

References

- Wikipedia:Near field communication

 (C) Copyright Nokia 2014. All rights reserved.