Workshop:
Android and NFC

Michael Roland
NFC Research Lab Hagenberg
University of Applied Sciences Upper Austria

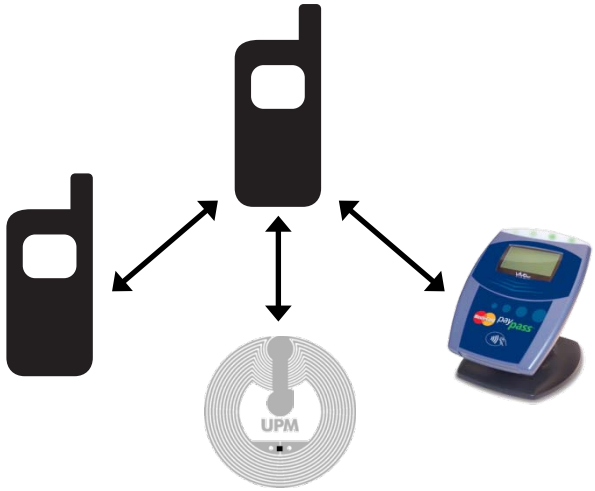
NFC Congress 2012
11 September 2012, Hagenberg, Austria

This work is part of the project "4EMOBILITY" within the EU program "Regionale Weltbewerbsfähigkeit OÖ 2007–2013 (Regio 13)" funded by the European regional development fund (ERDF) and the Province of Upper Austria (Land Oberösterreich).

Outline

- Basics
  - NFC
  - Tags
  - NDEF
- Android + NFC
- Hands-On
  - Part 1: NDEF Writer
  - Part 2: NDEF Reader
  - Part 3: Auto-start an app when a tag/NDEF record is detected

Basics: NFC's Operating Modes



- Peer-to-peer mode
  - Communication between two NFC devices
  - Android: Android Beam
- Reader/writer mode
  - NFC devices read & write NFC tags
  - ***Some*** NFC devices read & write other tags and smartcards
  - Android: Support for most NFC-compatible tags and smartcards
- Card emulation mode
  - NFC device imitates a contactless smartcard
  - Intended for security critical applications (payment …)
  - ***Not*** intended for emulation of NFC tags
    → Use peer-to-peer mode instead!
  - Android: No official support

Basics: NFC Tags

- NFC Forum defined 4 standard tag platforms
  - Type 1
    - Innovision Jewel/Topaz
  - Type 2
    - NXP MIFARE Ultralight / Ultralight C
    - NXP NTAG203
    - Infineon my-d move / my-d NFC
  - Type 3
    - Sony FeliCa
  - Type 4
    - NXP MIFARE DESFire
    - Implementable on any contactless smartcard with ISO 7816-4 support (e.g. on any JavaCard)

Basics: NFC Data Exchange Format (NDEF)

- Common format for …
  - storing data on NFC tags
  - transmitting data in peer-to-peer mode
- Data abstraction layer
  - Idea: Applications should work with any NFC tag platform
  - Advantage: Same API for reading and storing data on any NFC tag
    → Users can choose any tag platform (with sufficient storage)!
- Format
  - NDEF record is a data container with data type information
  - NDEF message is a series of one or more NDEF records
  - NFC tag contains NDEF message

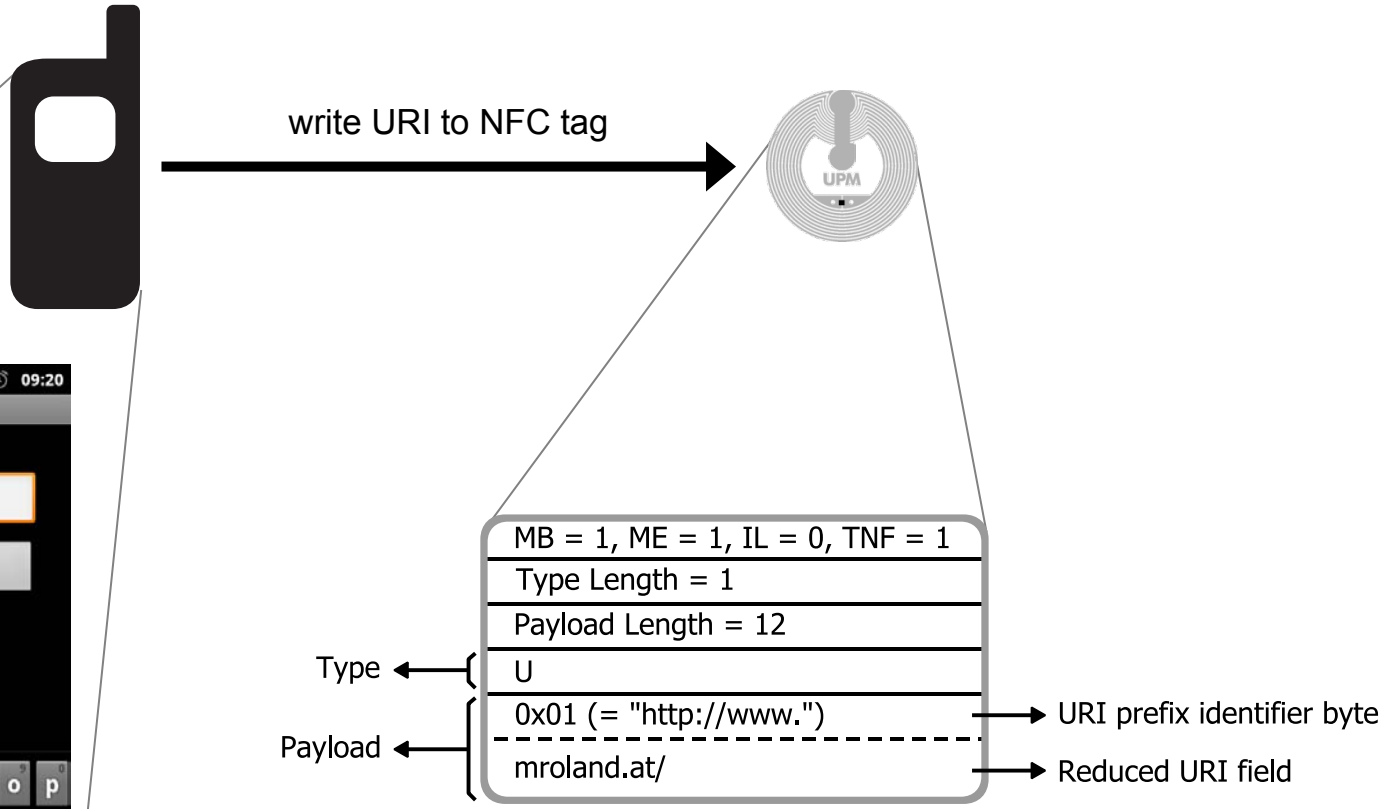
Basics: NDEF Records

- 4 groups of record types
  - NFC Forum Well-known types
    - Defined by the NFC Forum
    - E.g. URI (urn:nfc:wkt:**U**), Text (urn:nfc:wkt:**T**), SmartPoster (urn:nfc:wkt:**Sp**)
  - NFC Forum External types
    - Custom record types definable by application developers
    - E.g. urn:nfc:ext:**mroland.at:example**
  - MIME media types
    - Data formats defined by MIME media types
    - E.g. Business cards (**text/x-vcard**)
  - Absolute URI types
    - Data formats defined by URIs
    - Important: Don’t confuse this with the URI well-known type!

Android's NFC Features

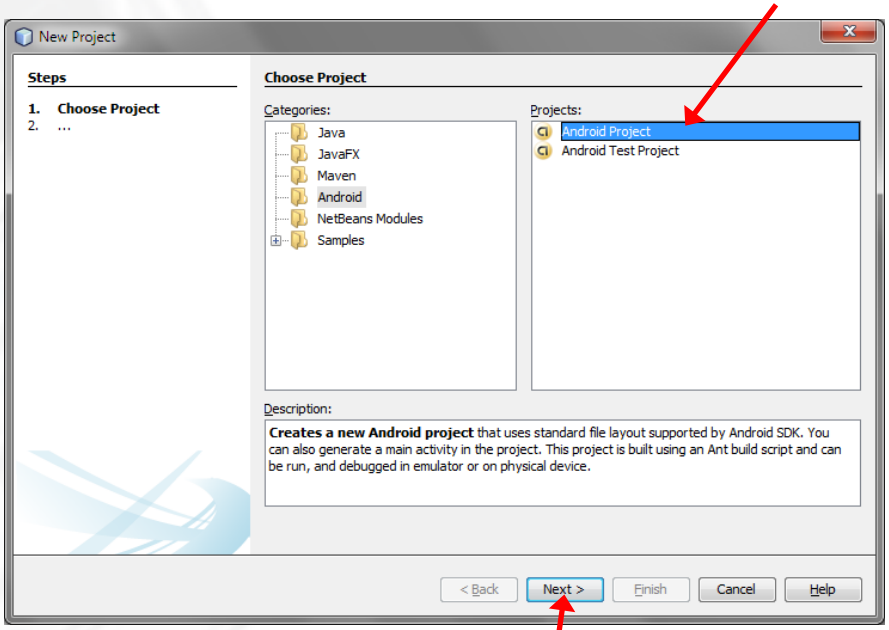
- Peer-to-peer mode
  - Android Beam: Transfer NDEF messages and (large) files
- Reader/writer mode
  - Read and write NDEF data on NFC tags
  - Communication with other tags and smartcards
    - ISO/IEC 14443-A based protocols
    - ISO/IEC 7816-4 APDUs
    - FeliCa (Lite)
    - ISO/IEC 15693 (only if NFC chipset supports it)
    - MIFARE Classic (only if NFC chipset supports it)
  - Auto start apps upon detection of
    - a certain NDEF record
    - a certain tag technology
    - tags not handled by other apps

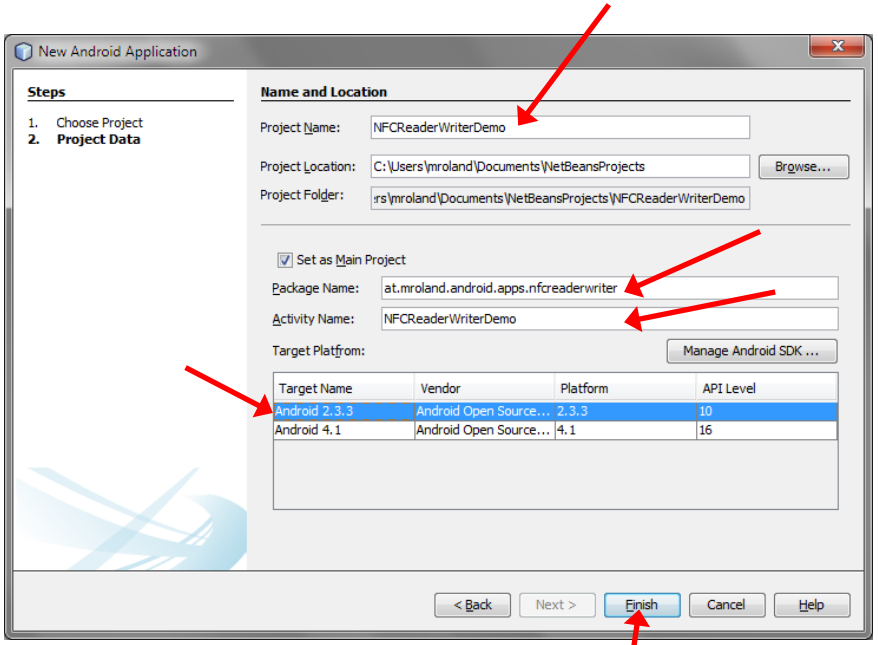
Hands-On (Part 1): NDEF Writer App



Step 1: Create a new Android Project

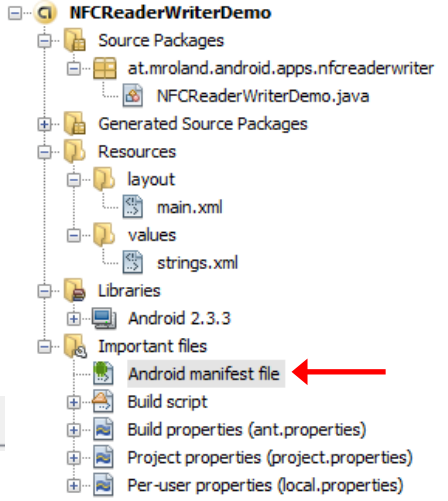
- File → New Project…





Step 2: Android Manifest & NFC

- Require Android 2.3.3 or later
- Request permission to use NFC
- Require devices with NFC hardware



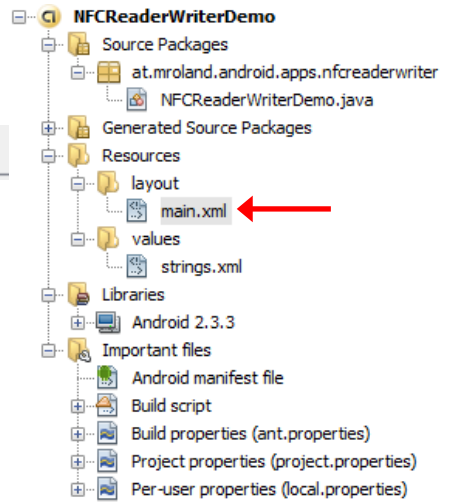
AndroidManifest.xml

```
<manifest ...>
    <!-- Require at least API level 10 (Android 2.3.3+): -->
    <uses-sdk android:minSdkVersion="10"
              android:targetSdkVersion="10" />
    <!-- Request permission to use NFC functionality: -->
    <uses-permission android:name="android.permission.NFC" />
    <!-- Restrict app to devices with NFC hardware: -->
    <uses-feature android:name="android.hardware.nfc"
                  android:required="true" />
    <application android:label="@string/app_name" >
```

Step 3: Add a User Interface

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent" android:layout_height="match_parent" >
    <LinearLayout android:orientation="vertical"
            android:layout_width="match_parent" android:layout_height="wrap_content" >

        <TextView android:layout_width="match_parent" android:layout_height="wrap_content"
            android:text="URL:"
            android:layout_marginLeft="30dp" android:layout_marginRight="30dp"
            android:layout_marginTop="15dp" android:layout_marginBottom="5dp" />

        <EditText android:id="@+id/myUrl"
            android:layout_width="match_parent" android:layout_height="wrap_content"
            android:text="http://www.mroland.at/"
            android:inputType="textUri"
            android:layout_marginLeft="30dp" android:layout_marginRight="30dp"
            android:layout_marginTop="0dp" android:layout_marginBottom="10dp" />

        <Button android:id="@+id/myWriteUrlButton"
            android:layout_width="match_parent" android:layout_height="wrap_content"
            android:text="Write to NFC Tag..."
            android:layout_marginLeft="30dp" android:layout_marginRight="30dp"
            android:layout_marginTop="0dp" android:layout_marginBottom="15dp"
            android:gravity="center" />

    </LinearLayout>
</ScrollView>
```

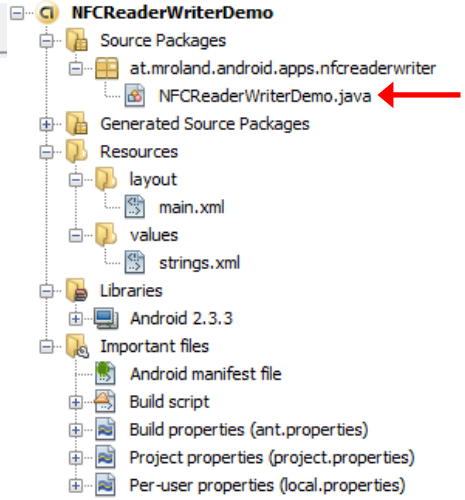


What we've got so far…

Step 4: Creating our Activity

NFCReaderWriterDemo.java

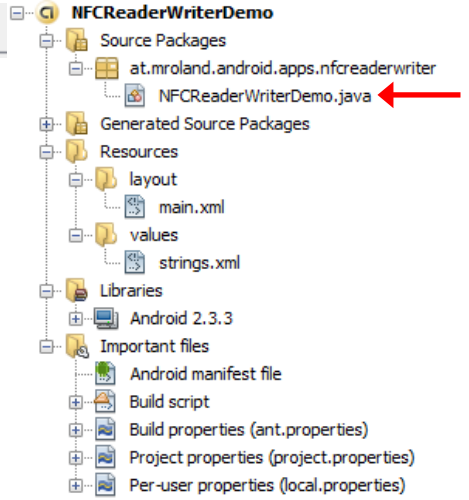
```
private static final int DIALOG_WRITE_URL = 1;
private EditText mMyUrl;
private Button mMyWriteUrlButton;
private boolean mWriteUrl = false;

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    mMyUrl = (EditText) findViewById(R.id.myUrl);
    mMyWriteUrlButton = (Button) findViewById(R.id.myWriteUrlButton);

    // Set action for "Write URL to tag..." button:
    mMyWriteUrlButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            mWriteUrl = true;
            NFCReaderWriterDemo.this.showDialog(DIALOG_WRITE_URL);
        }
    });
}
```



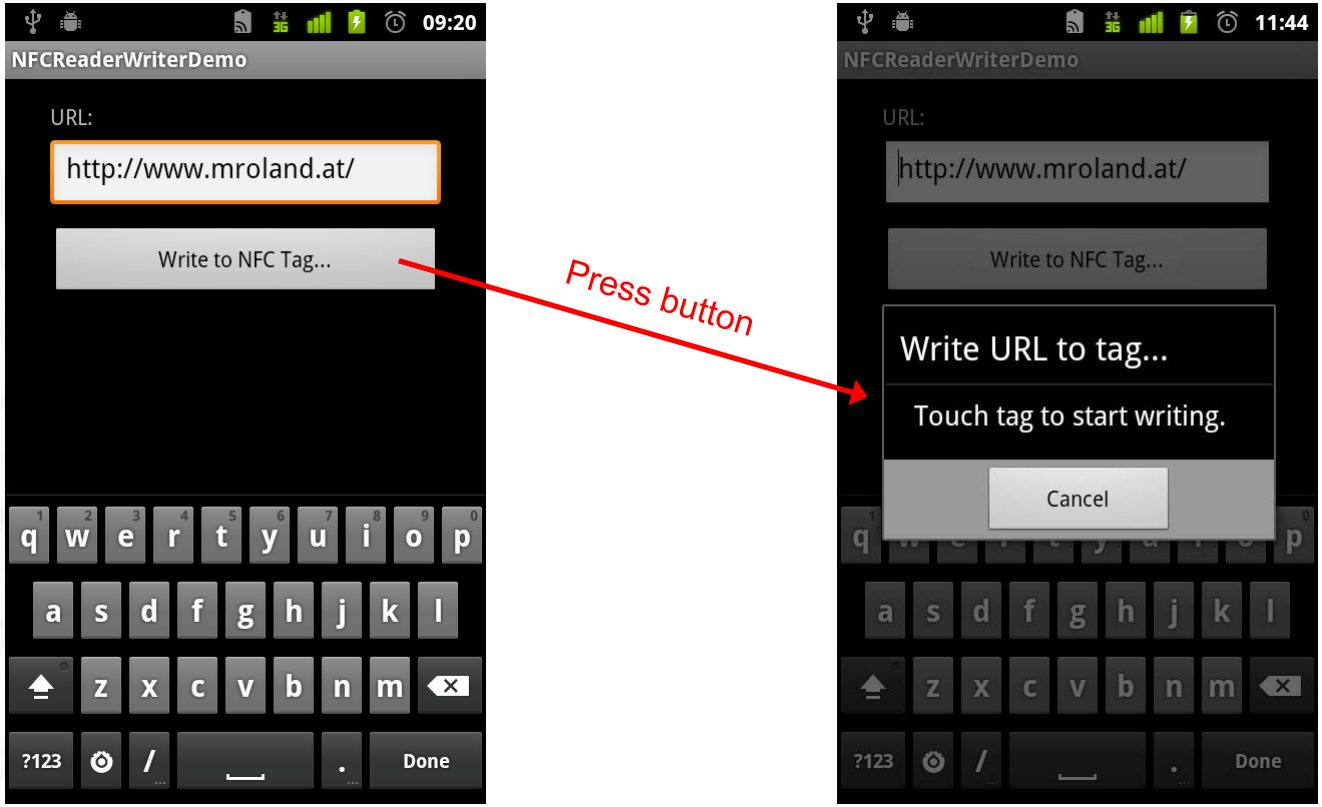
Step 5: A Dialog Box: Ready to Write to Tag

NFCReaderWriterDemo.java

```
@Override
protected Dialog onCreateDialog(int id, Bundle args) {
    switch (id) {
        case DIALOG_WRITE_URL:
            return new AlertDialog.Builder(this)
                    .setTitle("Write URL to tag...")
                    .setMessage("Touch tag to start writing.")
                    .setCancelable(true)
                    .setNeutralButton(android.R.string.cancel,
                                      new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface d, int arg) {
                            d.cancel();
                        }
                    })
                    .setOnCancelListener(new DialogInterface.OnCancelListener() {
                        public void onCancel(DialogInterface d) {
                            mWriteUrl = false;
                        }
                    }).create();
    }

    return null;
}
```



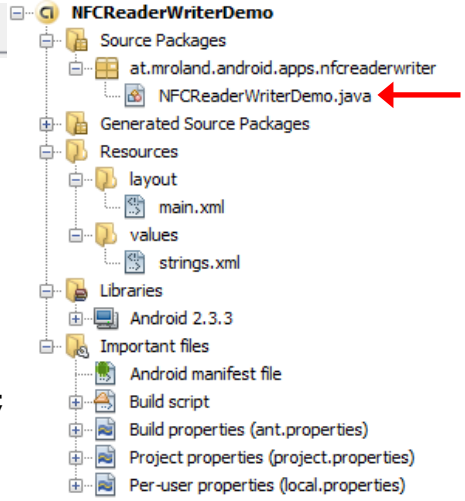
What we've got so far…



Step 6: Foreground Dispatch (Detect Tags)

NFCReaderWriterDemo.java

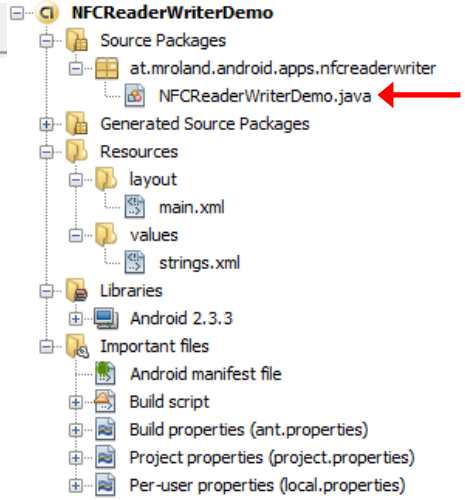
```
private static final int PENDING_INTENT_TECH_DISCOVERED = 1;
private NfcAdapter mNfcAdapter;

@Override
public void onResume() {
    super.onResume();

    // Retrieve an instance of the NfcAdapter:
    NfcManager nfcManager = (NfcManager) this.getSystemService(Context.NFC_SERVICE);
    mNfcAdapter = nfcManager.getDefaultAdapter();

    // Create a PendingIntent to handle discovery of Ndef and NdefFormatable tags:
    PendingIntent pi = createPendingResult(PENDING_INTENT_TECH_DISCOVERED, new Intent(), 0);

    // Enable foreground dispatch for Ndef and NdefFormatable tags:
    mNfcAdapter.enableForegroundDispatch(
            this,
            pi,
            new IntentFilter[]{ new IntentFilter(NfcAdapter.ACTION_TECH_DISCOVERED) },
            new String[][]{
                    new String[]{ "android.nfc.tech.NdefFormatable" },
                    new String[]{ "android.nfc.tech.Ndef" }
            });
}
```



Step 7: Cleanup Foreground Dispatch

NFCReaderWriterDemo.java

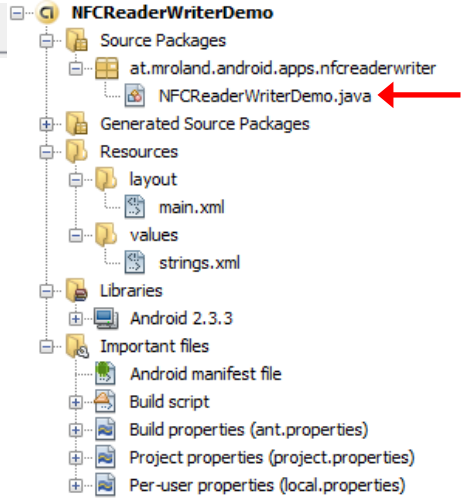
```
@Override
public void onPause() {
    super.onPause();

    // Disable foreground dispatch:
    mNfcAdapter.disableForegroundDispatch(this);
}
```



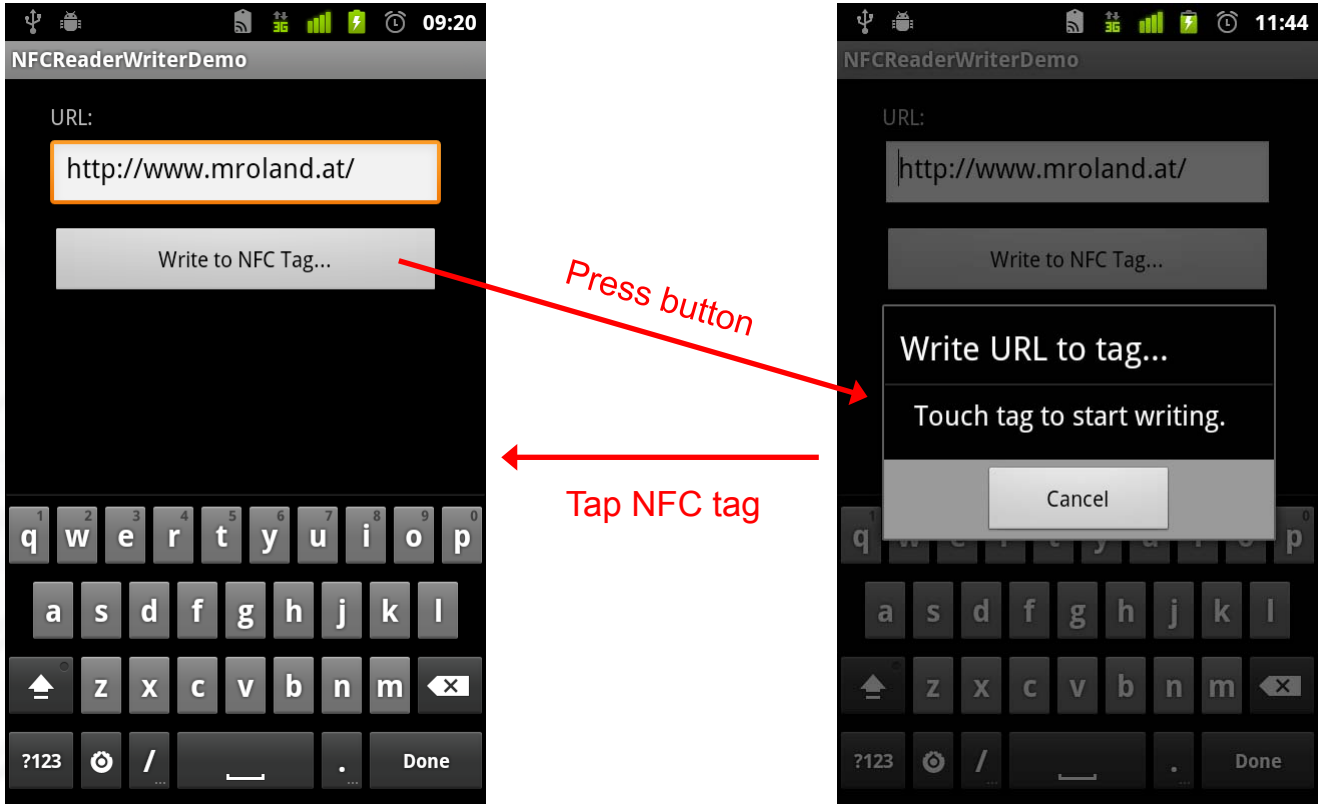
Step 8: Receive Foreground Dispatch Intent

NFCReaderWriterDemo.java

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode) {
        case PENDING_INTENT_TECH_DISCOVERED:
            resolveIntent(data, true);
            break;
    }
}

private void resolveIntent(Intent data, boolean foregroundDispatch) {
    String action = data.getAction();

    if (NfcAdapter.ACTION_TECH_DISCOVERED.equals(action)) {
        Tag tag = data.getParcelableExtra(NfcAdapter.EXTRA_TAG);

        if (foregroundDispatch && mWriteUrl) {
            // This is a foreground dispatch and we want to write our URL
            mWriteUrl = false;

            // TODO: Write the URL to the tag

            dismissDialog(DIALOG_WRITE_URL);
        }
    }
}
```



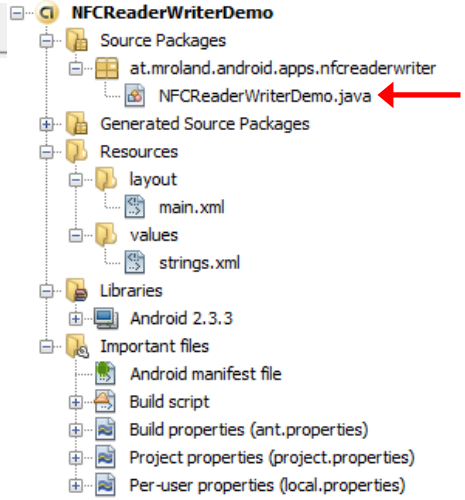
What we've got so far…



Step 9: Prepare our URI NDEF Message

NFCReaderWriterDemo.java

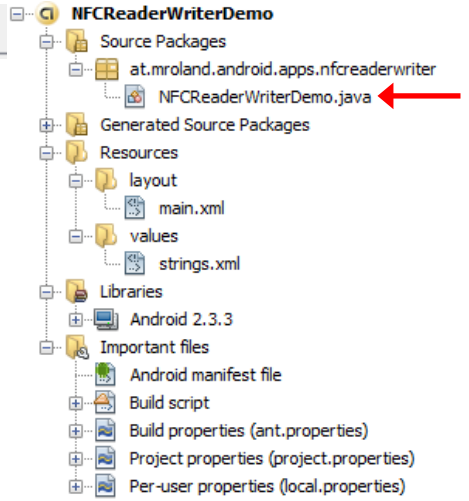
```
if (foregroundDispatch && mWriteUrl) {
    // This is a foreground dispatch and we want to write our URL
    mWriteUrl = false;

    // Get URL from text box:
    String urlStr = mMyUrl.getText().toString();

    // Convert to URL to byte array (UTF-8 encoded):
    byte[] urlBytes = urlStr.getBytes(Charset.forName("UTF-8"));

    // Assemble NDEF URI record payload:
    byte[] urlPayload = new byte[urlBytes.length + 1];
    urlPayload[0] = 0;  // no prefix reduction
    System.arraycopy(urlBytes, 0, urlPayload, 1, urlBytes.length);

    // Create a NDEF URI record (NFC Forum well-known type "urn:nfc:wkt:U")
    NdefRecord urlRecord = new NdefRecord(NdefRecord.TNF_WELL_KNOWN, /* TNF: NFC Forum well-known type */
                                          NdefRecord.RTD_URI,        /* Type: urn:nfc:wkt:U */
                                          new byte[0],               /* no ID */
                                          urlPayload);

    // Create NDEF message from URI record:
    NdefMessage msg = new NdefMessage(new NdefRecord[]{urlRecord});

    // TODO: Write the NDEF message to the tag
```



Step 10: Write NDEF to Preformatted Tag

NFCReaderWriterDemo.java

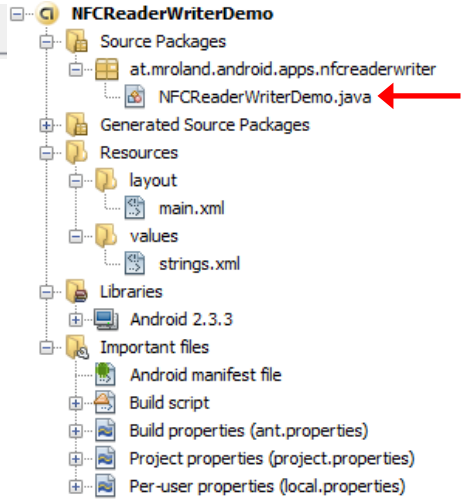
```
Tag tag = data.getParcelableExtra(NfcAdapter.EXTRA_TAG);
[...]
    // Create NDEF message from URI record:
    NdefMessage msg = new NdefMessage(new NdefRecord[]{urlRecord});

    Ndef ndefTag = Ndef.get(tag);
    if (ndefTag != null) {
        // Our tag is already formatted, we just need to write our message

        try {
            // Connect to tag:
            ndefTag.connect();

            // Write NDEF message:
            ndefTag.writeNdefMessage(msg);
        } catch (Exception e) {
        } finally {
            // Close connection:
            try { ndefTag.close(); } catch (Exception e) { }
        }
    } else {
        // Our tag is not NDEF formatted!
    }

    dismissDialog(DIALOG_WRITE_URL);
```



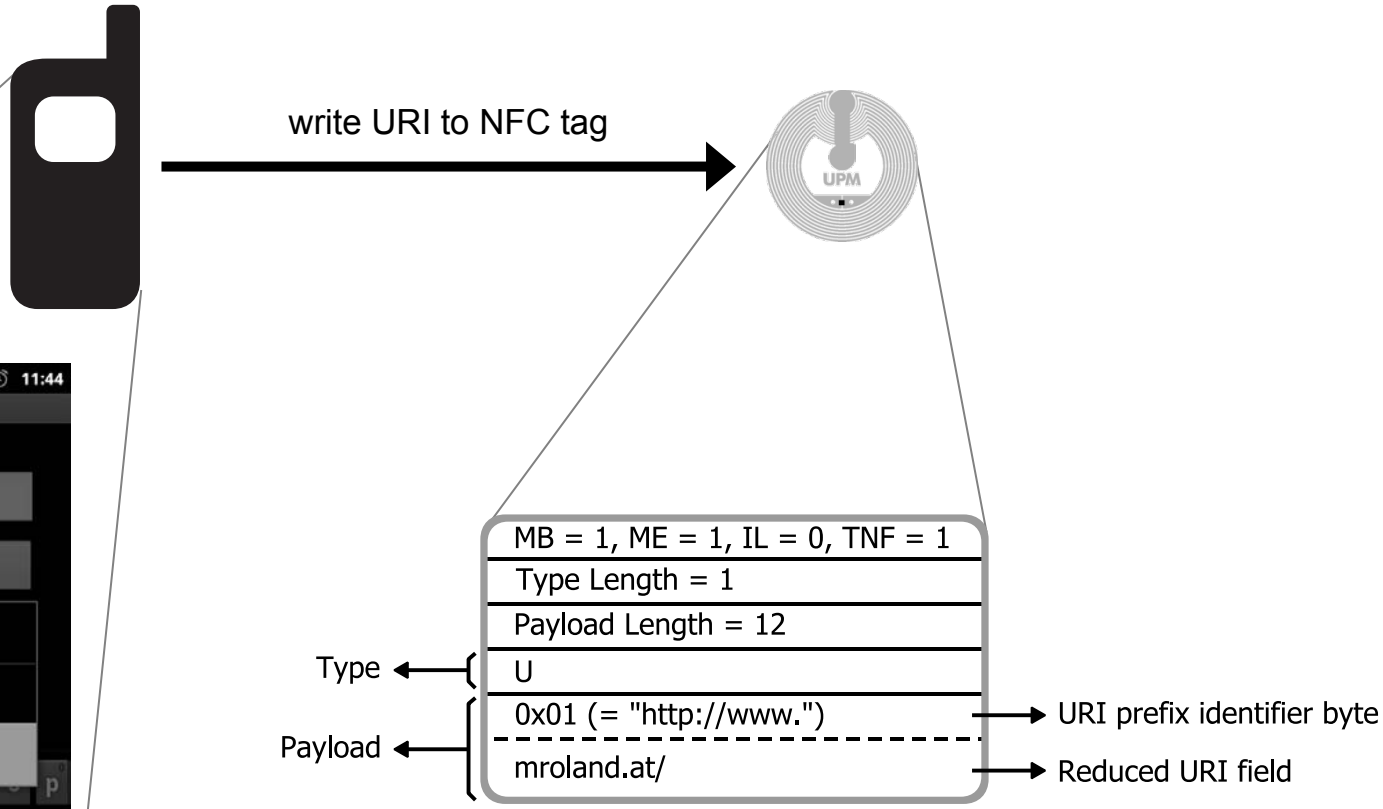
Step 11: Write NDEF to Unformatted Tag

NFCReaderWriterDemo.java

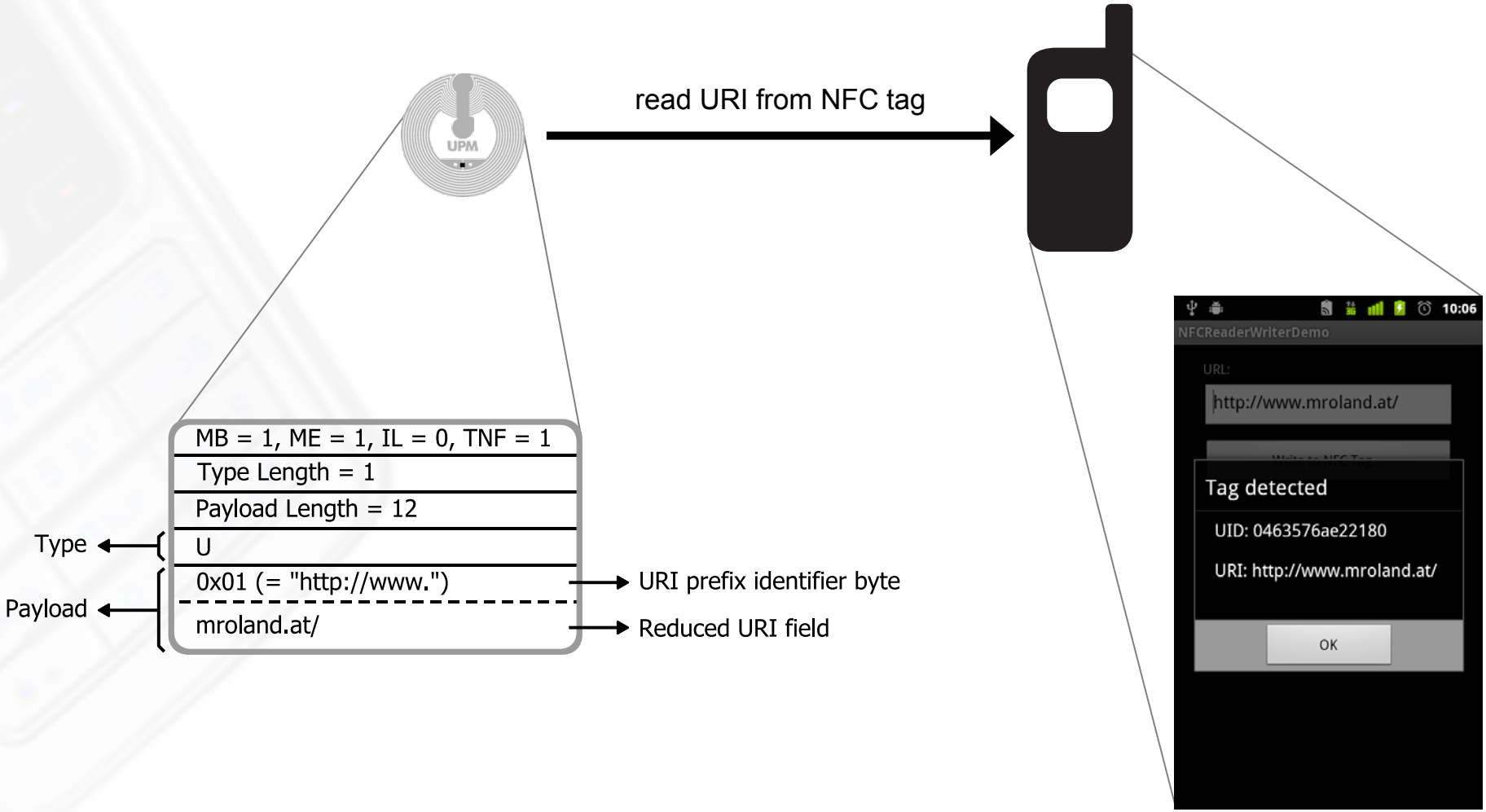
```
if (ndefTag != null) {
    [...]
} else {
    // Our tag is not NDEF formatted!
    NdefFormatable ndefFormatableTag = NdefFormatable.get(tag);
    if (ndefFormatableTag != null) {
        // Our tag is not yet formatted, we need to format it with our message

        try {
            // Connect to tag:
            ndefFormatableTag.connect();

            // Format with NDEF message:
            ndefFormatableTag.format(msg);
        } catch (Exception e) {
        } finally {
            // Close connection:
            try { ndefFormatableTag.close(); } catch (Exception e) { }
        }
    }
}

dismissDialog(DIALOG_WRITE_URL);
```



What we've got so far: NDEF Writer App



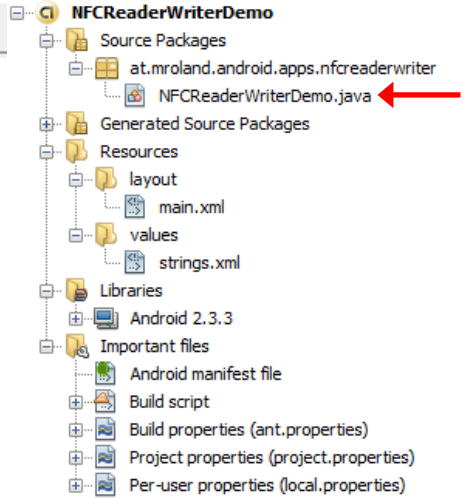
Hands-On (Part 2): NDEF Reader App



Step 1: A Simple Dialog Box: Detected a Tag

NFCReaderWriterDemo.java

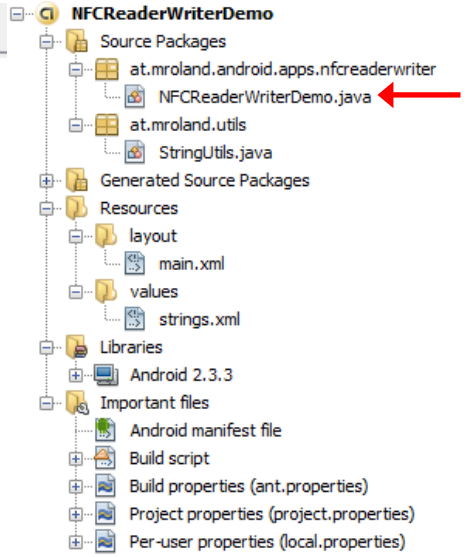
```
private static final int DIALOG_NEW_TAG = 3;
private static final String ARG_MESSAGE = "message";

protected Dialog onCreateDialog(int id, Bundle args) {
        [...]
        case DIALOG_NEW_TAG:
            return new AlertDialog.Builder(this)
                    .setTitle("Tag detected")
                    .setCancelable(true)
                    .setNeutralButton(android.R.string.ok,
                                      new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface d, int arg) {
                            d.dismiss();
                        }
                    }).create();

@Override
protected void onPrepareDialog(int id, Dialog dialog, Bundle args) {
    switch (id) {
        case DIALOG_NEW_TAG:
            String message = args.getString(ARG_MESSAGE);
            if (message != null) ((AlertDialog) dialog).setMessage(message);
            break;
    }
}
```



Step 2: Detect the Tag

NFCReaderWriterDemo.java

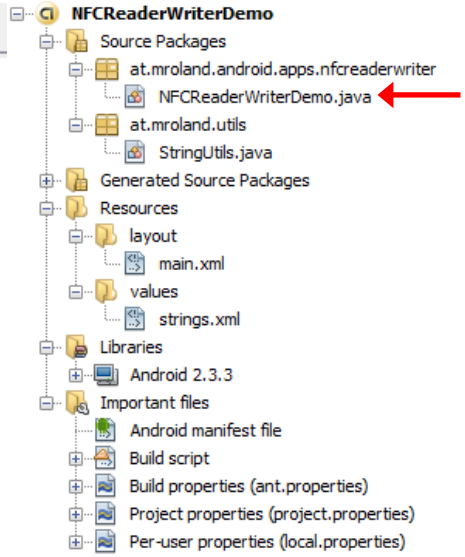
```
private void resolveIntent(Intent data, boolean foregroundDispatch) {
    String action = data.getAction();

    if (NfcAdapter.ACTION_TECH_DISCOVERED.equals(action)) {
        Tag tag = data.getParcelableExtra(NfcAdapter.EXTRA_TAG);

        if (foregroundDispatch && mWriteUrl) {
            [...]
        } else {
            // Let's read the tag whenever we are not in write mode

            // Retrieve information from tag and display it
            StringBuilder tagInfo = new StringBuilder();

            // Get tag's UID:
            byte[] uid = tag.getId();
            tagInfo.append("UID: ")
                   .append(StringUtils.convertByteArrayToHexString(uid))
                   .append("\n\n");

            // TODO: Read NDEF messages
        }
    }
}
```



Step 3: Read NDEF Message from Tag

NFCReaderWriterDemo.java

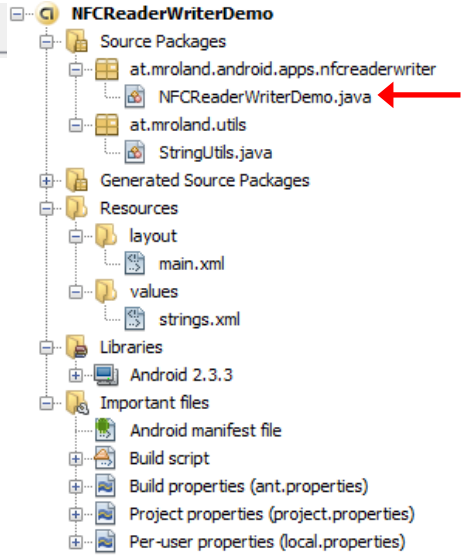
```
private void resolveIntent(Intent data, boolean foregroundDispatch) {
        [...]
        // Get tag's NDEF messages:
        Parcelable[] ndefRaw =
                data.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
        NdefMessage[] ndefMsgs = null;
        if (ndefRaw != null) {
            ndefMsgs = new NdefMessage[ndefRaw.length];
            for (int i = 0; i < ndefMsgs.length; ++i) {
                ndefMsgs[i] = (NdefMessage) ndefRaw[i];
            }
        }

        // TODO: Find our URI record
```



Step 4: Find URI Record in NDEF Messages

NFCReaderWriterDemo.java

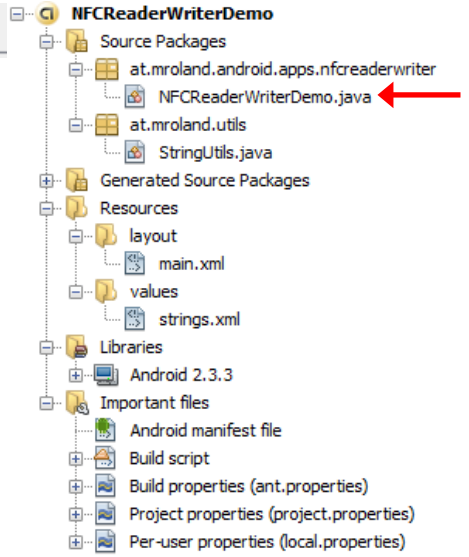
```
private void resolveIntent(Intent data, boolean foregroundDispatch) {
        [...]
        if (ndefMsgs != null) {
            // Iterate through all NDEF messages on the tag:
            for (int i = 0; i < ndefMsgs.length; ++i) {
                // Get NDEF message's records:
                NdefRecord[] records = ndefMsgs[i].getRecords();

                if (records != null) {
                    // Iterate through all NDEF records:
                    for (int j = 0; j < records.length; ++j) {
                        // Test if this record is a URI record:
                        if ((records[j].getTnf() == NdefRecord.TNF_WELL_KNOWN)
                                && Arrays.equals(records[j].getType(), NdefRecord.RTD_URI)) {
                            byte[] payload = records[j].getPayload();
                            // Drop prefix identifier byte and convert remaining URL to string (UTF-8):
                            String uri = new String(Arrays.copyOfRange(payload, 1, payload.length),
                                                    Charset.forName("UTF-8"));
                            tagInfo.append("URI: ").append(uri).append("\n");
                        }
                    }
                }
            }
        }
```



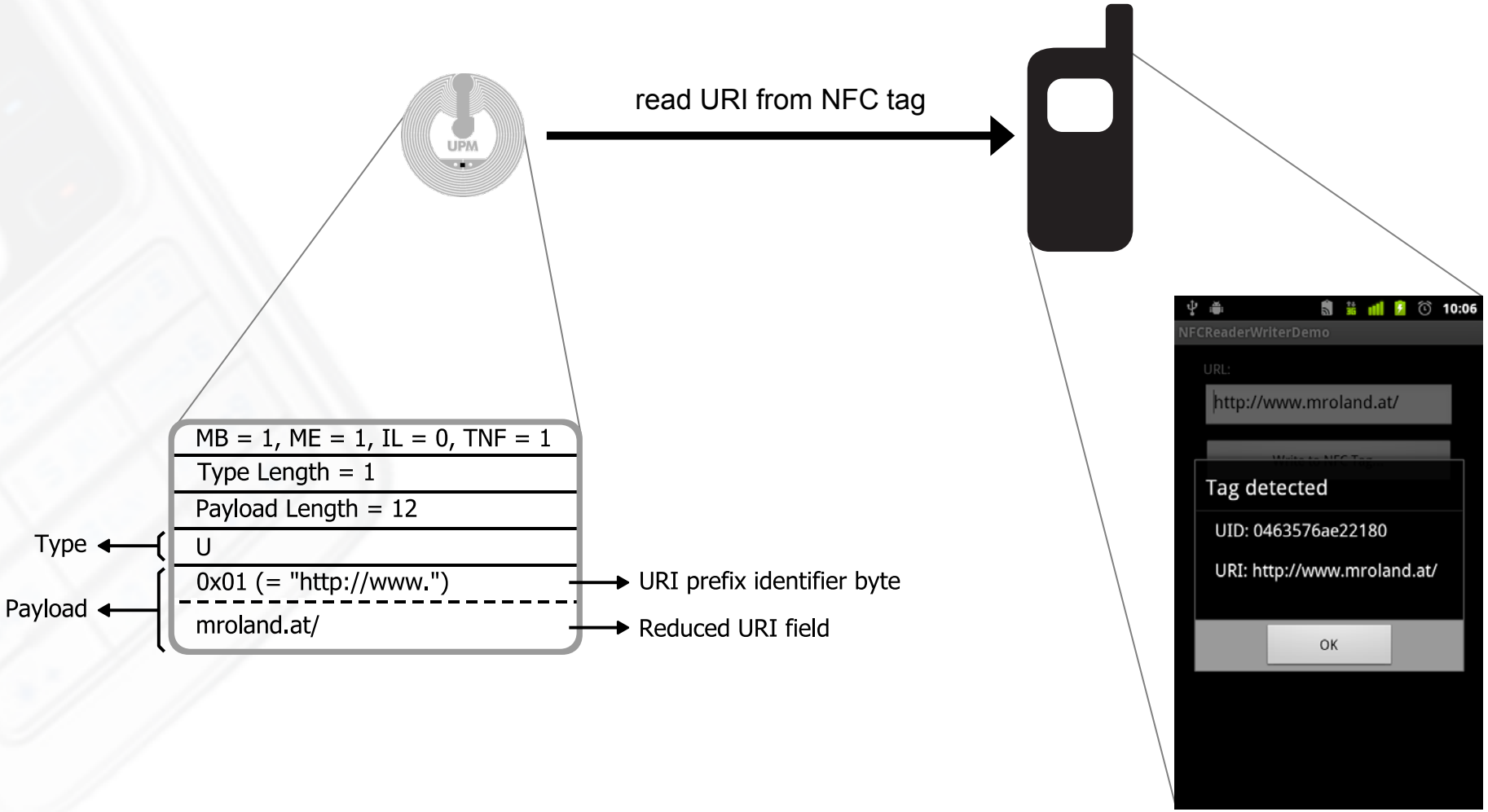
Step 5: Display Tag Data in a Dialog Box

NFCReaderWriterDemo.java

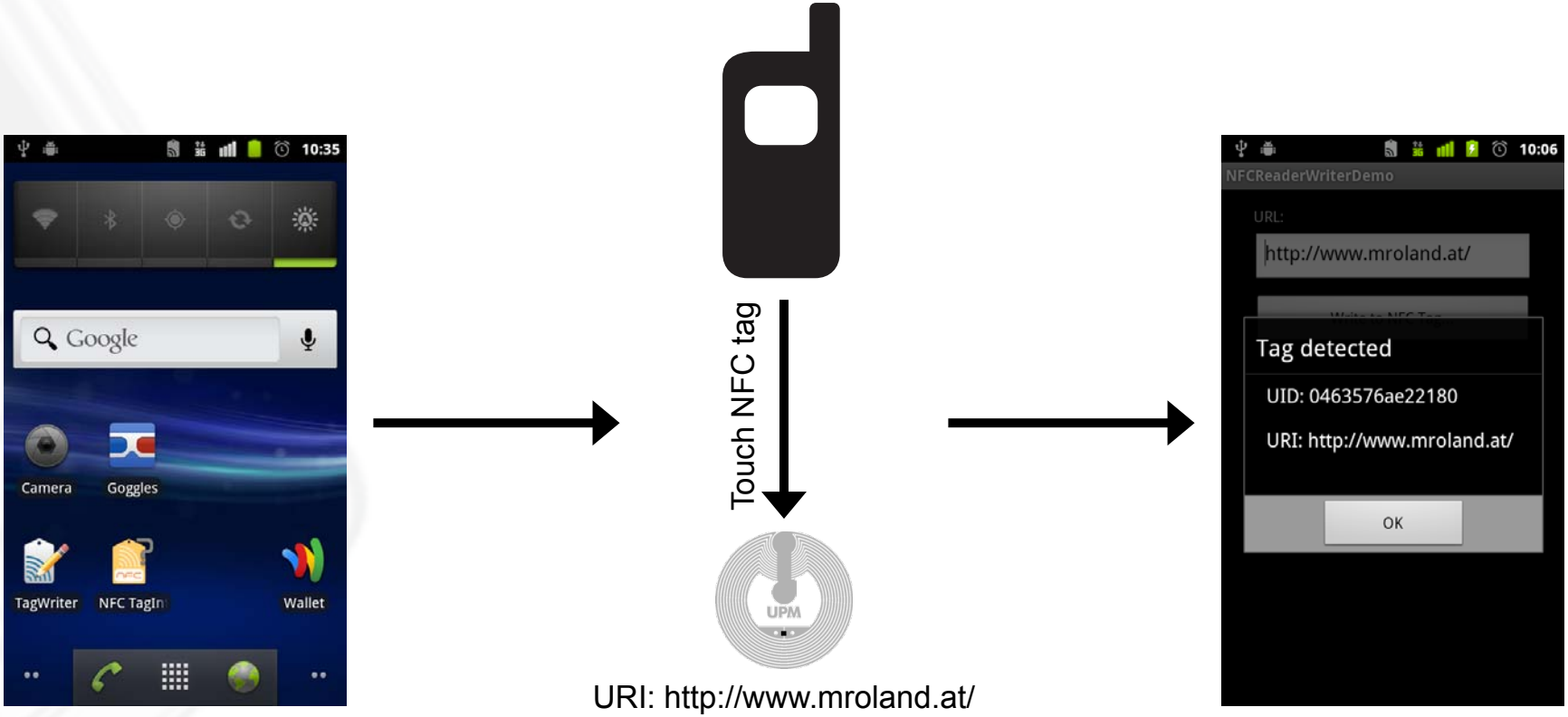
```
private void resolveIntent(Intent data, boolean foregroundDispatch) {
        [...]
        if (ndefMsgs != null) {
            [...]
        }

        Bundle args = new Bundle();
        args.putString(ARG_MESSAGE, tagInfo.toString());
        showDialog(DIALOG_NEW_TAG, args);
```



What we've got so far: NDEF Reader App



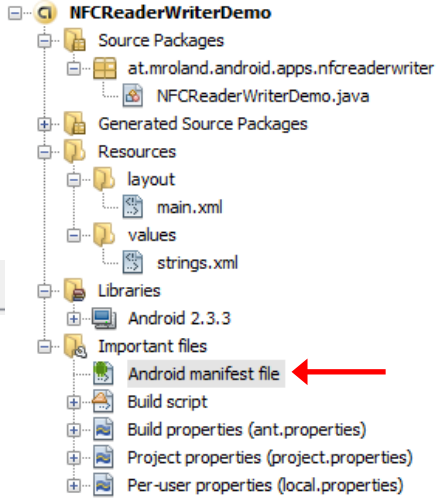
Hands-On (Part 3): Auto-start App for our URI



URI: http://www.mroland.at/

Step 1: Android Manifest & Auto-start on NFC Tag

- Launch Mode: *singleTask*
- Handling configuration changes
- Intent filter: NDEF_DISCOVERED



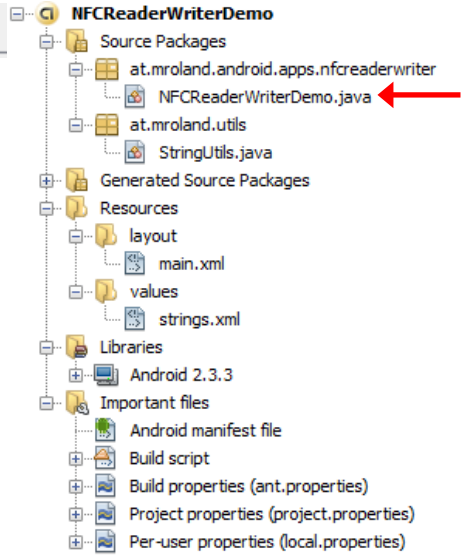
AndroidManifest.xml

```
<activity android:name="NFCReaderWriterDemo"
          android:label="@string/app_name"
          android:launchMode="singleTask"
          android:configChanges="orientation|keyboardHidden" >
    <!-- Make our app startable through the app launcher: -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <!-- Make our app startable through our URL: -->
    <intent-filter>
        <action android:name="android.nfc.action.NDEF_DISCOVERED" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:scheme="http"
              android:host="www.mroland.at"
              android:pathPrefix="" />
    </intent-filter>
</activity>
```

Step 2: Handle NDEF_DISCOVERED Intent

NFCReaderWriterDemo.java

```
@Override
public void onCreate(Bundle savedInstanceState) {
        [...]
    // Resolve the intent that started us:
    resolveIntent(this.getIntent(), false);
}

@Override
public void onNewIntent(Intent data) {
    resolveIntent(data, false);
}

private void resolveIntent(Intent data, boolean foregroundDispatch) {
    String action = data.getAction();

    // We were started from the recent applications history: just show our main activity
    if ((data.getFlags() & Intent.FLAG_ACTIVITY_LAUNCHED_FROM_HISTORY) != 0) return;

    if (NfcAdapter.ACTION_TECH_DISCOVERED.equals(action)
            || NfcAdapter.ACTION_NDEF_DISCOVERED.equals(action)) {
        Tag tag = data.getParcelableExtra(NfcAdapter.EXTRA_TAG);

        if (foregroundDispatch && mWriteUrl) {
```



Downloads

- Full example code:

  https://www.mroland.at/uploads/2012/09/NFCCongress2012_NFCReaderWriterDemo_full.zip

Michael Roland, MSc
Research Associate, NFC Research Lab Hagenberg
University of Applied Sciences Upper Austria

Mail: michael.roland (at) fh-hagenberg.at

Web: www.mroland.at

This work is part of the project "4EMOBILITY" within the EU program "Regionale Wettbewerbsfähigkeit OÖ 2007–2013 (Regio 13)" funded by the European regional development fund (ERDF) and the Province of Upper Austria (Land Oberösterreich).